JUHISED PROGRAMMEERIMISEKS ARVUTILE

EC-1022 KEELES A S S E M B L E R

TARTU RIIKLIK ÜLIKOOL
Arvutuskeskus

JUHISED PROGRAMMEERIMISEKS ARVUTILE EC-1022 KEELES ASSEMBLER

Koostanud: A.Jaeger
Ü.Kaasik

TARTU 1977

Kinnitatud Matemaatikateaduskonna
nõukogus 27. juunil 1977. a.

Käesolev metoodiline juhend on kavandatud progremmeerijatele, kes kasutavad progammeerimiskeelt ASSEMBLER arvutil EC-1022. Niisugust eesmärki ning väljaande rangelt piiratud mahtu arvestades ongi selgitused esitatud üsna napisõnaliselt ja piirdutud üpris väikese arvu näidetega. Et eesti keeles pole seni veel ilmunud ülevaadet kolmanda põlvkonna arvutite operatsioonisüsteemidest, siis sissejuhatusena tutvustatakse kõigepealt operatsioonisüsteemi DOS mõningaid põhimõisteid.

1. Operatsioonisüsteemist DOS

Operatsioonisüsteem DOS (= Disk Operating System) võimaldab konkreetsete ülesannete lahendamiseks määratud programmide koostamisel kasutada järgmisi lähtekeeli: FORTRAN, RPG, COBOL, PL/1 ja ASSEMBLER. Mingis lähtekeeles kirjutatud programmi nimetatakse lähtemooduliks ehk lähteprogrammiks, tema transleerimisel saadakse nn. objektmoodul. Et mistahes sisendkeelest saadud objektmooduleid võib komplekteerida enamasti iga konkreetse täitmisele mineva nn. probleemprogrammi koosseisu, siis sisaldab objektmoodul lisaks oma sisule veel mitmesugust informatsiooni nii moodulitevaheliste seoste organiseerimiseks kui ka täitmise asukohast mälus sõltuvate programmiobjektide modifitseerimiseks.

Iga täitmisele minev objektmoodul tuleb tingimata komplekteerida (ka siis kui tema juurde mingeid teisi mooduleid ei lülitata), mille tulemusena saadakse täitmiseks valmis absoluutmoodul ehk faas. Absoluutmoodul tuuakse vajaduse korral täitmiseks sisemälu sellesse kohta, kuhu ta on komplekteeritud. Kui soovitakse programmi täita sisemälu mingis teises kohas, siis tuleb vastav objektmoodul üldiselt uuesti komplekteerida. Täiendavaid võtteid rakendades ning süsteemi poolt seatud kitsendusi arvestades võib programmeerija siis-

ki vajaduse korral koostada ka nn. ujuprogrammi, mis komplekteeritakse küll mälu kindlasse kohta, kuid mida saab täita ka mujal mälus. Selleks peab aga programm ise sisaldama programmilõigu, milles modifitseeritakse kõik programmi konkreetsest asukohast sõltuvad programmiobjektid.

Operatsioonisüsteemis DOS on ettevalmistamise eri staadiumis olevate moodulite säilitamiseks olemas kolm eraldi teeki. Lähtemoodulid säilitatakse üldiselt nn. lähtemoodulite ehk S-teegis (= Souroe library), objektmoodulid objektmoodulite ehk R-teegis (= Relocatable library) ning absoluutmoodulid vastavalt absoluutmoodulite ehk C-teegis (= Core-image library). Seejuures on operatsioonisüsteemi DOS kõikides versioonides S- ja R-teekide korral lubatud ka erateegid, C-teekide korral aga alates versioonist DOS 2.0 .

C-teegis paiknev, täitmiseks sisemällu lugemist ootav programm võib koosneda ühest või ka mitmest faasist. Seejuures võib programmil olla kas liht-, ülekatte- või siis põhifaasiga ülekattestruktuur.

Lihtstruktuuriga programm koosneb ainult ühest faasist, mis täitmiseks tuuakse sisemällu tervikuna.

Ülekattestruktuuriga programm koosneb mitmest faasist, kusjuures iga järjekordne faas kutsub täitmiseks sisemällu järgmise faasi nii, et see kas tervikuna või vähemalt osaliselt katab eelmise.

Põhifaasiga ülekattestruktuuri korral paikneb üks nn. põhifaas programmi kogu täitmise aja sisemälus, ülejäänud faasid aga tuuakse sisemällu vajaduse korral, kusjuures nende omavahelise struktuuri määrab programmeerija.

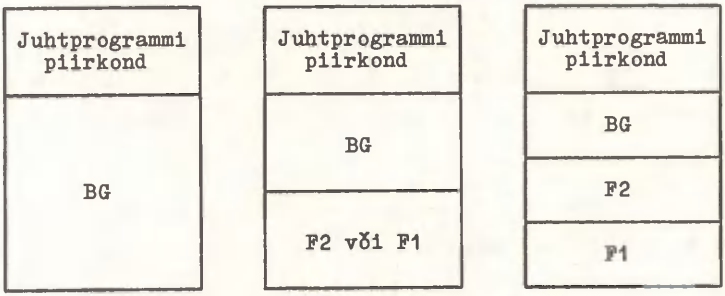
Operatsioonisüsteem DOS võimaldab programme täita nii ühe programmi režiimis kui ka nn. multikasutamise režiimis, mille korral saab paralleelselt täita kuni kolme ülesannet. Seejuures tuleb iga ülesanne spetsiaalsete juhtlausete abil vormistada nn. tööna - selliselt vormistatud ülesanded kujutavad endast juhtlausete (ning tarbe korral ka vajalike algandmete) jadasid, mis koondatakse tööde sisendvooks ülesannete nn. pakktöötluse organiseerimiseks.

Soovi korral võib töö jaotada veel nn. sammudeks, näidates vastavate juhtlausetega iga sammu täitmise jaoks vajaliku programmi. Nii näiteks on silumisjärgus oleva programmi korral tavalisteks sammudeks lähteteksti korrigeerimine (TEEKERi koosseisu kuuluva programmiga MAINT), transleerimine (ASSEMBLERi korral programmiga ASSEMBLY või ASSEMBLF), komplekteerimine (programmiga LNKEDT) ning silumine (programmiga ATLEDT).

Ressursside jaotamiseks paralleelselt täidetavate programmide vahel jagatakse süsteemi juhtprogrammi piirkonnast üle jääv sisemälu osa nn. jagudeks. Üht jagudest nimetatakse seejuures taustaks (lühend BG = BaokGround), ülejäänuid aga frontideks (lühend F = Foreground). Sellest lähtudes nimetatakse ka täidetavaid ülesandeid vastavalt taust- või frontülesanneteks.

Operatsioonisüsteem DOS võimaldab probleemprogrammide tarvis üle jäävat sisemälu osa vaadelda kas üheainsa jaona (selleks jaoks on tingimata BG), kahe jaona (üheks jaoks on ikka BG, teiseks aga kas F2 või F1), või siis kolme jaona (jaod BG, F2 ja F1). Arvuti kogu sisemälu jaotus täitmisel

olevate programmide vahel toimub seega ühe variandi kohaselt järgmise kolme skeemiga näidatud võimaluste hulgast (ka jagude järjekord on selline):



Multikasutamise režiimi korral on üldiselt kõikides jagudes võimalik teostada tööde pakktöötlust, kuid süsteemi lähteolekus (pärast algpaigaldamist) on selleks kohe valmis ainult taust. Pakktöötluse frontides käivitab vastavalt vajadustele ja võimalustele operaator spetsiaalse korralduse abil.

Mitme programmi paralleelsel täitmisel lähtub operatsioonisüsteem protsessori aja jaotamisel programmi prioriteedist. Nimelt on kõige kõrgema prioriteediga esimeses frondis (F1) täidetav programm ning kõige madalamaga taustprogramm. Iga kord, kui mingis jaos täidetav programm peab ootama näiteks sisend-väljundoperatsiooni lõppu, annab süsteem juhtimise prioriteedile vastavalt teisele täitmiseks valmis olevale programmile (kui selline leidub). Seejuures saab kõrgema prioriteediga programm juhtimise tagasi vahetult pärast tema poolt käivitatud sisend-väljundoperatsiooni lõppu.

Niisugust juhtimise üleandmise põhimõtet arvestades on soovitatav anda kõrge prioriteet (s.t. täita frondis) rohkesti sisend-väljundoperatsioone sisaldavatele programmidele, suure arvutusmahuga programmid aga täita taustas.

Operatsioonisüsteem koosneb väga paljudest programmidest, millised oma funktsioonidelt võib jaotada juhtprogrammiks, infovahetussüsteemiks, translaatoriteks ning teenindusprogrammideks. Juhtprogramm, mis kindlustab kogu arvuti töö automaatse juhtimise, koosneb omakorda järgmisest neljast programmist.

Süsteemi töö alustamisel esimesena täidetavaks programmiks on alati ALGPAIGALDAJA, mis teostab mitmesugused ettevalmistavad protseduurid ning mille täitmise tulemusena on süsteem valmis tööde vastuvõtmiseks. Ettevalmistavate tööde käigus loeb ALGPAIGALDAJA kettalt SYSRES (millel asub kogu operatsioonisüsteem) sisemällu ka juhtprogrammi koosseisu kuuluva programmi SUPERVIISOR, õigemini selle tähtsaima, süsteemi kogu töötamise aja sisemälus asuva osa, nn. tuuma.

SUPERVIISORi tuum sisaldab selliseid arvuti töö juhtimisel kõige sagedamini vaja minevaid programme nagu näiteks paigaldaja, välisseadmete töö juhtimiseks ette nähtud nn. kanalite planeerija, arvuti ja programmide, aga samuti ka kettaseadme tõrgete töötlemise ning mälukaitse ja aja-anduri teenindamise programme. Peale nende kuulub SUPERVIISORi koosseisu veel terve rida suhteliselt vähem kasutamist leidvaid programme (operaatoriga sidepidamiseks, välisseadmete tõrgete töötlemiseks, kontrollpunktide moodustamiseks jne.),

mis püsivalt asuvad seadmel SYSRES ning vajaduse korral loetakse sealt juhtprogrammi piirkonda kuuluvasse nn. transiitpiirkonda (vastavaid programme nimetatakse ka SUPERVIISORi transiitmooduliteks).

Tööde järjestikust täitmisele võtmist organiseerib juhtprogrammi koosseisu kuuluv TÖÖDE JUHTIJA, mille süsteemi töö alustamisel (hiljem aga iga töö lõpetamise järel) loeb sisemällu (probleemprogrammide piirkonna vastavasse jakku) SUPERVIISOR. Väljaspool tööde üldist sisendvoogu antavate n.ö. üksikprogrammide täitmisele võtmist organiseerib viimane neljast juhtprogrammi koosseisu kuuluvast programmist - ÜKSIKPROGRAMMIDE KÄIVITAJA .

Operatsioonisüsteemis DOS leiab laialdast kasutamist nn. makrosüsteem, mis annab programmeerijale vahendid ASSEMBLERi võimaluste laiendamiseks. Makrosüsteemi abil on operatsioonisüsteemis realiseeritud näiteks kogu infovahetussüsteem ning mitmesugused SUPERVIISORi poole pöördumisega seotud protseduurid.

Peale selliste n.ö. süsteemi makrokäskude võib aga programmeerija juurde luua ja kasutada järjest uusi temale vajalikke makrokäske. Makrosüsteemiga seotud põhimõisteteks on mäkrokäsk, makromäärang ning makrolaiend.

Makrokäsk on selline ASSEMBLERi laiendamiseks juurde loodud keelekonstruktsioon, mille spetsiaalne programm (nn. makrogeneraator) asendab lähteprogrammis ASSEMBLER-programmi teatava lõiguga. Seejuures on makrokäsu koodiks uue keelekonstruktsiooni nimi, aadressosaks aga vajaduse korral tege-

like parameetrite loetelu (leheküljel 13 toodud programmis esinevad näiteks süsteemi makrokäsud GET, PUT, EOJ, DTFCD, DTFPR ning PRMOD).

Makrosüsteemi nõuete kohaselt vormistatud programmilõiku, millega makrogeneraator makrokäsu asendab, nimetatakse makromääranguks. See sisaldab peale ASSEMBLERi direktiivide veel spetsiaalseid direktiive, mille abil toimub ühelt poolt makromäärangu vormistamine (direktiivid MACRO, MEND ja makrokäsu koodi ning parameetrite nimesid määrav nn. prototüübidirektiiv), parameetrite kirjeldamine ja nende väärtuse muutmine (direktiivid GBL, LCL, SET) ning tarbe korral teadete trükkimine (direktiiv MNOTE), teiselt poolt aga vahelepandava programmilõigu enda moodustamise (genereerimise) juhtimine (direktiivid AIF, AGO, ACTR).

Makrogeneraatori poolt töödeldud ning lähteprogrammi vastavasse kohta vahele pandud programmilõiku nimetatakse makrolaiendiks. Makrolaiend sisaldab juba ainult ASSEMBLERi direktiive, kus makromäärangus kasutatud formaalsed parameetrid on asendatud nende tegelike (makrokäsus näidatud) väärtustega.

Makrosüsteemi kasutamise teeb eriti lihtsaks see, et makrogeneraator on lülitatud ASSEMBLER-translaatori koosseisu. Iga lähteprogrammi töötlebki kõigepealt makrogeneraator, mille töö tulemusena saadud n.ö. "puhas" ASSEMBLER-programm transleeritakse tavalises korras. Translaator teostab muidugi ka kõikide süsteemi makrokäskude töötlemise (nendele vastavad makromäärangud asuvad juba valmis kujul lähtemoodulite teegis).

Operatsioonisüsteemide üheks iseloomulikumaks omaduseks on nende genereeritavus, s.t. võimalus moodustada kasutajate vajadustele ja arvuti tehnilistele näitajatele kõige enam vastav operatsioonisüsteem. Muutuvatele tingimustele paindlikuma reageerimise võimaldamiseks on süsteemide loomisel aluseks võetud nn. moodulprintsiip, mis lubab mitmesugustest süsteemi võimalusi realiseerivatest moodulitest kokku panna operatsioonisüsteemide erinevaid variante (moodulprintsiibi järjekindla kasutamise muudab suhteliselt lihtsaks arvuti makrosüsteem).

Arvutiga kaasa antav juba eelnevalt genereeritud operatsioonisüsteemi variant peab muidugi silmas teatavaid keskmisi süsteemile esitatavaid nõudeid ning ka arvuti standardset tehnilist varustust, andes seega piisavad võimalused arvuti kasutamiseks, aga samuti ka operatsioonisüsteemi uute variantide genereerimiseks.

Süsteemi genereerimise esimeseks etapiks on alati tema planeerimine, millega määratakse kindlaks SUPERVIISORile esitatavad nõuded (milliseid protseduure ta peab täitma jms.) ning samuti ka süsteemsete teekide maht ja sisu.

Planeeritud SUPERVIISOR kirjeldatakse vastavate makrokäskude abil ning saadud makrokäskude jada transleeritakse ASSEMBLER-translaatoriga, objektmoodul komplekteeritakse ja valmis SUPERVIISOR kataloogitakse tavalisel viisil absoluutmoodulite teeki.

Lõpuks toimub uute süsteemsete teekide moodustamine, mille järel operatsioonisüsteemi järjekordne variant ongi tööks valmis.

2. ASSEMBLER-programm

Keeles ASSEMBLER kirjutatud programm ehk ASSEMBLER-programm kujutab endast nn. direktiivide jada. Direktiivide kirjutamiseks kasutatav alfabeet sisaldab järgmises tabelis loetletud 51 põhisümbolit:

Tähed	Numbrid	Eraldajad
A B C D E P G H I J K L M N O P Q R S T U V W X Y Z ¤ # @	Ø 1 2 3 4 5 6 7 8 9	␣ = + - * / ( ) . , ' &

(tuleb tähele panna, et null kirjutatakse kujul Ø ning tühik kujul ␣; sümbolid ¤, # ja @ on loetud tähtede hulka). Peale nende tohib aga tekstikonstantides ja kommentaarides kasutada ka koodi ДКОИ (vt. Programme kõigile XII, lisa 2) teisi sümboleid.

Tähtedel A, B, C, D, E ja F võib mõningates konstruktsioonides olla ka numbri tähendus. Nimelt kui keele süntaksi kohaselt on tegemist kuueteistkümnendsüsteemi arvuga, siis neid tähti tõlgendatakse selle süsteemi täiendavate numbrimärkidena - vastavalt (kümnend-)arvude 10, 11, 12, 13, 14 ja 15 tähistena.

Direktiiv koosneb üldiselt tehte koodist ja aadressosast. Peale selle võib aga direktiivi tarbe korral märgendada ning kirjutada aadressosa järele märkuse (mis transleerimisele ei kuulu). Kõik need direktiivi osad peavad olema üksteisest eraldatud vähemalt ühe tühikuga.

ASSEMBLER-programm kirjutatakse spetsiaalsetele blankettidele (vt. lk. 13). Iga blanketileht varustatakse programmi ja koostaja nimedega, šifriga ning lehe järjekorranumbriga, kuid seda informatsiooni ei perforeerita. Perforeerimisele kuuluvad blanketi järgnevate ridade positsioonidesse 1 - 72 kirjutatavad direktiivid ja positsioonidesse 73 - 80 kirjutatavad identifikaatorid (mis võivad ka puududa). Identifikaatoreid kasutatakse programmi identifitseerimiseks ja /või direktiivide nummerdamiseks.

Direktiivid kirjutatakse blanketile enamasti nn. standardformaadis, kuid vajaduse korral võib formaati ka muuta, teatades sellest translaatorile direktiivi ICTL (vt. lk. 74) abil. Standardformaadi korral on direktiivi algusposition 1, kust alates märgendatud direktiivide korral tuleb kirjutada etikett (etikett on tähega algav tähtedest ja numbritest koosnev ülimalt kaheksasümboliline sõne). Märgendamata direktiivides jäävad esimesed üheksa positsiooni täitmata.

Tehte koodi kirjutamist alustatakse blanketil positsioonist 10. Tehte koodina kasutatakse ASSEMBLERis ülimalt viietähelist mnemoonilist koodi, mis kujutab endast lühendit selle operatsiooni ingliskeelsest nimetusest. Kõigi käesolevas väljaandes vaadeldavate direktiivide mnemoonilised koodid koos vastavate ingliskeelsete fraasidega on toodud lisas 2 (vt. lk. 83-87), kus ühtlasi viidatakse iga direktiivi kirjeldust sisaldavale leheküljele. Tuleb arvestada, et erandina koodide moodustamise üldreeglist kasutatakse ujukoma arvudega seotud direktiivide mnemoonilistes koodides

TRÜ AK

ASSEMBLER
EC-1022

PROGRAMM *Informatsiooni trükkimine perfokaartidelt*

KOOSTAS *A. Jaeger*

ŠIFFER *B-342* LEHT *1*

MÄRGEND	KOOD	OPERANDID	J	IDENTIF.
	START			
	BALR	2,Ø		
	USING	*,2		
GET	GET	CARD		
	MVC	Ø(8Ø,3),Ø(1Ø)		
	PUT	PRINT		
	B	GET		
EOFCD	EOJ			
CARD	DTFCD	IOAREA1=IO1CD,IOAREA2=IO2CD,IOREG=(1Ø),,EOFADDR=EOFCD,	X	
		DEVADDR=SYSIPT		
PRINT	DTFPR	BLKSIZE=8Ø,CONTROL=YES,IOAREA1=RIDA1,,IOAREA2=RIDA2,	X	
		IOREG=(3),,MODNAME=PRMOD,DEVADDR=SYSLST		
PRMOD	PRMCD	CONTROL=YES,IOAREA2=YES		
	DS	ØH		
IO1CD	DS	CL8Ø		
IO2CD	DS	CL8Ø		
RIDA1	DS	CL8Ø		
RIDA2	DS	CL8Ø		
	END			

tähte E (= Exponential) või D (= Double precision), kuigi lisas 2 vastav ingliskeelne fraas sisaldab selle asemel sõna "short" või "long". Teise erandina kasutatakse mõnedes koodides täheühendi UD asemel ühtainust tähte W (= Double U).

Direktiivi aadressosa kirjutamist tuleb blanketil alustada positsioonist 16. Aadressosa sisaldab üldiselt ühe või mitu (enamasti kaks) operandi - viimasel juhul eraldatakse operandid üksteisest komadega. Mõne direktiivi korral võivad aga operandid ka hoopis puududa.

Märkus (mis võib sisaldada koodi ДКОИ kõiki sümboleid) kirjutatakse vajaduse korral aadressosa järele, jättes vahele vähemalt ühe tühiku. Pikema, kogu rida hõivava märkuse kirjutamiseks võib kasutada spetsiaalset nn. kommentaardirektiivi, mille tunnuseks on sümbol * blanketi algusposit-sioonis.

Mistahes direktiivi kirjutamist võib jätkata kuni positsioonini 71 (lõpp-positsioon standardformaadi korral). Kui direktiiv ei mahu ühele reale, siis võib tema kirjutamist jätkata järgmisel real alates positsioonist 16 (jätkamis-positsioon standardformaadi korral), kusjuures eelmise rea positsiooni 72 tuleb jätkamise tunnusena tingimata kirjutada mingi tühikust erinev sümbol.

Tihti esineda kippuvate vigade vältimiseks tuleb mitmele reale ulatuvate direktiivide kirjutamisel arvestada, et kui jätkatava rea kirjutamine lõpetatakse enne positsiooni 71, siis viimased tühikud muudavad järgmisel real oleva jätku märkuseks (ja direktiivi seega reeglina vigaseks). Tühikud jätkatava rea viimastes positsioonides ei tähenda märkuse

algust ainult siis, kui viimaseks sümboliks enne neid tühikuid on operande eraldav koma (seda võimalust on kahel korral kasutatud ka leheküljel 13 toodud näiteprogrammis).

Blankettidele kirjutatud ASSEMBLER-programm perforeeritakse kaartidele - iga rida eraldi kaardile. Saadud kaardipakile tuleb aga lisada veel operatsioonisüsteemile esitatava tellimuse vormistamiseks vajalikud nn. juhtkaardid. Nii näiteks tüüpiline tellimus ASSEMBLER-programmi nn. kontrolltransleerimiseks saadakse viie juhtkaardi lisamise teel, millega kogu kaardipakk omandab kuju:

```
//␣JOB␣B342
//␣OPTION␣NOXREF
//␣EXEC␣ASSEMBLF
(kaardid ASSEMBLER-programmi tekstiga)
/*
/&
```

ASSEMBLER-programmi transleerimise käigus trükitakse ühtlasi välja nn. transleerimisprotokoll, mis sisaldab välisetikettide tabeli, lähtemooduli koos saadud objektmooduliga ja aadresskonstantide ning etikettide tabelid (vt. lk. 16, kus toodud transleerimisprotokolli näide on küll tehnilistel põhjustel veidi moonutatud). Kui translaator avastab programmis vea, siis vigasele direktiivile järgnevasse ritta trükitakse vea tunnusena ***ERROR*** ja väljastatakse teade vea kohta. Võimalike veateadete (ingliskeelsed) tekstid koos vastavate vigade tõenäoliste põhjuste selgitustega on toodud lisas 1 (vt. lk. 76-82).

```
// JOB B342                                          ØØ.1Ø.36
// EXEC ASSEMBLF

                    EXTERNAL SYMBOL DICTIONARY                PAGE 1
SYMBOL  TYPE ID  ADDR   LENGTH LD ID
         PC  Ø1 ØØØØØØ ØØØØ14

                                                              PAGE 1
  LOC   OBJECT CODE  ADDR1 ADDR2 STMT  SOURCE STATEMENT
ØØØØØØ  Ø52Ø                      1          BALR  2,Ø
ØØØØØ2                            2          USING *,2
ØØØØØ2  ØØØØ ØØØØ          ØØØØØ  3          L     1Ø,B
         ***ERROR***
ØØØØØ6  5ØBØ 2ØØE          ØØØ1Ø  4          ST    11,A
                                  5          EOJ
                                  6+*DOS/EC EOJ   V.M 2.Ø
ØØØØØA  ØAØE                      7+         SVC   14
ØØØØØC  ØØ1Ø                      8 Y        DC    Y(A)
ØØØØ1Ø                            9 A        DS    F
                                 1Ø          END
                   RELOCATION DICTIONARY                  PAGE 1
 POS.ID  REL.ID  FLAGS  ADDRESS              27/Ø6/77
   Ø1      Ø1      Ø4    ØØØØØC

                        CROSS-REFERENCE                   PAGE 1
SYMBOL   LEN  VALUE  DEFN    REFERENCES     27/Ø6/77
A        ØØØØ4 ØØØØ1Ø ØØØØ9  ØØØ4  ØØØ8
B        ****UNDEFINED*****  ØØØ3
Y        ØØØØ2 ØØØØØC ØØØØ8

                          DIAGNOSTICS                     PAGE 1
STMT  ERROR CODE  MESSAGE                    27/Ø6/77
      IJYØ46      AT LEAST ONE RELOCATABLE Y-TYPE CONSTANT
   3  IJYØ24      NEAR OPERAND COLUMN 4--UNDEFINED  SYMBOL
   1 STATEMENT FLAGGED IN THIS ASSEMBLY

EOJ B342                          ØØ.12.Ø3,DURATION ØØ.Ø1.27
```

3. Aadresside moodustamine

Arvuti sisemälu koosneb 8-bitilistest baitidest, mis on nummerdatud (adresseeritud) alates nullist. Direktiivides opereeritakse tavaliselt mäluväljadega, mille pikkus võib olla 1 ÷ 256 baiti. Välja aadressina kasutatakse tema esimese (vasakpoolseima) baidi aadressi, välja pikkus aga kas näidatakse vastava direktiivi aadressosas otseselt, või siis määrab selle näiteks vastaval väljal paikneva konstandi pikkus.

Adresseerimise lihtsustamise huvides leiavad tihti kasutamist veel järgmised standardse pikkusega väljad: poolsõna (pikkus kaks baiti, aadress peab jaguma kahega), sõna (pikkus neli baiti, aadress peab jaguma neljaga) ning topeltsõna (pikkus kaheksa baiti, aadress peab jaguma kaheksaga).

Direktiiviga määratud operatsioonist osa võtva mäluvälja tegelik aadress A ehk absoluutaadress moodustatakse alati vähemalt kahe komponendi - baasaadressi e. baasi ja nihke e. suhtaadressi liitmise teel:

$$A = (B) + D ,$$

kus (B) tähendab nn. baasregistris B paiknevat baasaadressi, D aga suhtaadressi.

Et suhtaadressi maksimaalne lubatud väärtus on 4095 (arvuti käsus on tema jaoks ruumi ainult 12 bitti), siis kujutab baas endast 4096-baidise mälubloki algusaadressi. Baasi maksimaalseks väärtuseks võib põhimõtteliselt olla 16777215 ($= 2^{24}-1$), kuid konkreetse arvuti korral ei tohi baasaadress

muidugi ületada sisemälu maksimaalset aadressi.

Nimetatud kahele komponendile lisandub mõnedes direktiivides veel kolmas - indeks, ning absoluutaadress moodustatakse niisugustes direktiivides kujul

$$A = (X) + (B) + D ,$$

kus (X) tähistab nn. indeksregistris X asuvat indeksit (indeksi maksimaalne lubatud väärtus on sama, mis baasil). Kui suhtaadress annab kauguse baasile vastava mälubloki algusest, siis indeks võimaldab järjestikust pöördumist ülejäänud kahe komponendiga määratud massiivi elementide poole.

Baas- ja indeksregistrid pole spetsiaalselt selleks otstarbeks konstrueeritud registrid, vaid kujutavad endast arvuti nn. üldregistreid, milledest mõnede kasutamise baas- või indeksregistrina määrab programmeerija. Üldregistreid (igaüks pikkusega 4 baiti) on arvutis kokku kuusteist (registrinumbritega Ø,1,2,...,15) ning peale siinnimetatud otstarbe kasutatakse neid veel summaatorina fikskoma ning loogiliste operatsioonide teostamisel, tsükliloendajatena jne.

Lisaks üldregistritele on arvutis veel nn. ujukomaregistrid, millede põhiotstarbe reedab juba nende nimetus. Ujukomaregistreid on neli (registrinumbritega Ø, 2, 4 ja 6), igaüks nendest pikkusega 8 baiti (= 64 bitti).

ASSEMBLERi direktiivi aadressossa kirjutatavad suhtaadressid, baas- ja indeksregistrite numbrid vms. kujutavad endast üldjuhul avaldisi, mille väärtused määratakse transleerimisel. Avaldis koosneb kas ühest või mitmest üksliikmest e. termist, kusjuures termiks võib olla kas konstant, koha-

loenduri väärtus, etikett, etiketi pikkusatribuut või literaal. Vastavalt sellele, kas termi väärtus sõltub programmi asukohast mälus või mitte, jagunevad termid veel suhtelisteks ja absoluutseteks.

Konstantterm on selline absoluutterm, mille väärtus näidatakse temas endas kas kümnend-, kuueteistkümnend- või kahendarvuna, või siis sümbolitena. Seda väärtust (täpsemalt, selle väärtuse kahendekvivalenti) kasutab translaator direktiivile vastava käsu konstrueerimisel. Konstanttermi kasutatakse peamiselt selliste elementide näitamiseks direktiivides, nagu vahetu operand, mask, nii suht- kui ka absoluutaadress, mäluvälja pikkus jne., kusjuures konstanttermi võimalikest tüüpidest valitakse tavaliselt see, mille abil vajalikku elementi on kõige mugavam esitada.

Kõige enam kasutamist leidev kümnendterm kirjutatakse tavalise märgita kümnendarvuna, mille väärtus ei tohi olla suurem kui 16 777 215. Kõikide ülejäänud konstanttermide kirjutamist alustatakse tüüpi määrava tunnusega (X = kuueteistkümnendterm, B = kahendterm, C = sümbolterm), millele apostroofide vahel järgneb kas märgita arv vastavas arvusüsteemis või siis tüübi C korral sümbolite jada. Seejuures kuueteistkümnendtermi maksimaalseks lubatud väärtuseks on X'FFFFFF' , kahendtermi maksimaalseks lubatud pikkuseks 24 bitti, sümbolterm võib aga sisaldada ainult kuni kolm (koodi ДКОИ) sümbolit. Sümboltermi kirjutamisel peab veel silmas pidama seda, et eritähendust omavate sümbolite ' ja & lülitamiseks termi tuleb nad tingimata kirjutada kahekordselt (näiteks sümbolitele '&' vastav sümbolterm tuleb kirjutada

kujul C'''&&''').

Lähteprogrammi transleerimisel kasutab translaator nn. kohaloendurit, mille jooksvaks väärtuseks on järjekordse direktiivi alguse suhtaadress (absoluutaadressid selguvad alles programmi paigaldamisel mällu). Kohaloenduri algväärtuseks võetakse kas null või programmeerija poolt direktiiviga START (vt. lk. 42) näidatud väärtus.

Iga direktiivi transleerimise järel suurendab translaator kohaloendurit saadud konstruktsiooni (käsk, reserveeritud mäluväli, konstant jne.) poolt hõivatud baitide arvu võrra. Kui direktiiv on märgendatud, siis omistab translaator vastavale etiketile kohaloenduri jooksva väärtuse, mida hiljem kasutab nende direktiivide transleerimisel, mille aadressosas see etikett esineb.

Kohaloenduri jooksvat väärtust saab programmeerija kasutada suhttermina sümboli * abil, mis tähistab seega sellesama direktiivi suhtaadressi, mille aadressosas ta esineb.

Programmi mitmesuguste elementide (konstandid, aadressid jne.) tähistamiseks kasutatakse ASSEMBLER-programmis etikette. Etikette valib programmeerija üldiselt oma suva järgi, hoolitsedes vaid selle eest, et erinevaid elemente tähistatakse erinevate etikettidega. Segaduste vältimiseks pole siiski soovitatav kasutada programmis sümbolitega IJ algavaid etikette, aga samuti ka niisuguseid etikette, mille esimesed sümbolid langevad kokku mingi selles programmis vaja mineva faili nimega, sest sellise struktuuriga etikette kasutab infovahetussüsteem.

Programmi mingi elemendi tähistamiseks tuleb vastava

etiketiga märgendada seda elementi kirjeldav direktiiv. Mitmesuguste programmielementide kirjeldamiseks sisaldab vaadeldav keel terve rea spetsiaalseid direktiive, nagu näiteks konstantide moodustamise ja mälu reserveerimise direktiivid, mille ette kirjutatud etikett tähistab vastavalt kas konstandi või siis reserveeritud mäluvälja esimese (vasakpoolseima) baidi aadressi. Samuti võib aga märgendada ka ASSEMBLER-programmi mistahes teisi direktiive (näiteks kasutamiseks aadressidena juhtimisdirektiivides).

Tähistatavate programmielementide iseloomust sõltuvalt jagunevad etiketid suhtelisteks ja absoluutseteks. Programmi asukohast mälus sõltuvate elementide (konstandid, direktiivid, millele soovitakse anda juhtimist jne.) tähistamiseks kasutatavaid etikette nimetatakse suhtetikettideks, kindlaid väärtusi omavate elementide (registrite numbrid jne.) tähistamiseks kasutatavaid etikette aga absoluutetikettideks.

Üldiselt peavad kõik ASSEMBLER-programmi direktiivide aadressosades kasutatavad etiketid olema selles programmis määratud, s.t. esinema mingi direktiivi märgendina. Etiketi määramine omakorda tähendab, et translaator omistab sellele etiketile väärtuse, pikkusatribuudi ja suhtelisuse tunnuse (näitab, kas tegemist absoluut- või suhtetiketiga). Reserveeritud mäluosa, direktiivi, konstanti jne. tähistava etiketi väärtuseks on vastavat elementi sisaldava mäluvälja esimese baidi aadress (kohaloenduri jooksev väärtus), üldregistrit tähistava etiketi väärtuseks aga näiteks tema number. Etiketi võimalikud väärtused on $0 \div 16\ 777\ 215_{10}$.

Etiketi pikkusatribuudi väärtuseks on üldiselt selle

etiketiga märgendatud direktiivis määratud mäluvälja pikkus baitides (kui näiteks etikett on nelja baidi pikkuse käsu märgendiks, siis etiketi pikkusatribuudi väärtuseks on neli). Erinevalt sellest reeglist omistab translaator direktiivide START, CSECT, DSECT, EXTRN ja LTORG märgendamiseks kasutatavate etikettide pikkusatribuudile väärtuse üks. Etiketi pikkusatribuudi väärtust võib programmeerija kasutada absoluuttermina, kirjutades kõigepealt tunnusena sümbolid L' ning seejärel vastava etiketi.

Termi viimasena vaadeldav liik, nn. literaal annab programmeerijale võimaluse vajaliku konstandi näitamiseks otse direktiivi aadressosas. Direktiivi transleerimisel paigutab translaator literaalina esitatud konstandi programmi spetsiaalsesse nn. literaalide piirkonda, transleerimise tulemusena saadava käsu aadressossa aga formeerib vastava konstandi aadressi. See ongi oluliseks erinevuseks literaali ja konstanttermi vahel, sest viimasega määratud konstanti kasutab translaator otse moodustatava käsu mingi komponendina. Peale selle pakub literaal konstanttermiga võrreldes laialdasemaid võimalusi konstantide esitamiseks.

Literaali kirjutamine erineb konstandi kirjutamisest direktiivi DC (vt. lk. 27-38) aadressosas põhiliselt vaid selle poolest, et vahetult konstandi ette tuleb tingimata panna literaali tunnusena sümbol = . Märgime veel seda, et kuna literaal asendatakse käsus vastava aadressiga, siis tuleb teda lugeda suhttermide hulka.

Avaldiste moodustamiseks võib kasutada kõiki vaadeldud termiliike, silmas tuleb aga pidada järgmisi näpunäiteid.

Termide aritmeetilise kombinatsiooni moodustamisel on lubatud kasutada nelja aritmeetilist tehet, tehtemärkideks on:

\+ (liitmine),
\- (lahutamine),
* (korrutamine) ja
/ (jagamine).

Peale selle võib tavaliste reeglite järgi kasutada ümarsulge, kuid avaldis ei tohi sisaldada üle viie paari sulge üksteise sees. Avaldis ei tohi alata aritmeetilise tehte märgiga ega sisaldada üle kuueteistkümne termi. Vigasteks loetakse avaldised, mis sisaldavad kaks järjestikust termi või tehtemärki, samuti niisugused, kus toimub korrutamis- või jagamistehe suhttermidega.

Avaldistena võib kasutada näiteks järgmisi konstruktsioone:

```
A+*-X'D3'
ALG+X'3D'
=C'TEKSTIKONSTANT'
*+1Ø
AB+L'A
=P'36'
A-AB/(1Ø-PK*L'C)
A1+(B3-8*12)
```

Avaldise väärtuse arvutamisel asendatakse kõigepealt termid nende väärtustega, seejärel teostatakse vajalikud aritmeetikatehted (korrutamine ja jagamine enne liitmist ja

lahutamist), kusjuures suluavaldised arvutatakse eelisjärjekorras (mitmekordsete sulgude korral seestpoolt väljapoole). Jagamise tulemuseks on alati täisarv (murdosa jäetakse lihtsalt ära). Seejuures on lubatud ka nulliga jagamine, mille tulemuseks võetakse null.

Avaldiste lõppväärtused peavad olema -2^{24} ja $2^{24}-1$ vahel (negatiivne väärtus kujutatakse täiendkoodis), vahetulemused aga -2^{31} ja $2^{31}-1$ vahel. Üksnes A-tüüpi aadresskonstantides võib ka avaldise lõppväärtus olla -2^{31} ja $2^{31}-1$ vahel.

Igale avaldisele omistatakse tinglikult ka pikkusatribuut (seda läheb tarvis näiteks direktiivi EQU märgendina kasutatava etiketi pikkusatribuudi määramiseks või välja pikkuse näitamiseks mõnedes arvutikäskudes), kusjuures selleks võetakse avaldise esimese (või ainukese) termi pikkusatribuut. Seejuures avaldises esineva etiketi pikkusatribuudi väärtuse (L'<etikett>) või konstanttermi pikkusatribuut loetakse võrdseks ühega, literaali pikkusatribuudiks võetakse tema poolt määratud konstandi pikkus, kohaloenduri väärtuse (*) pikkusatribuudiks aga selle käsu pikkus, mille aadressosas ta esineb.

Nagu avaldiste koosseisu kuuluvad termid, nii ka avaldised ise jagunevad suhtelisteks ja absoluutseteks vastavalt sellele, kas nende väärtus sõltub programmi asukohast mälus või mitte. Seejuures võib absoluutavaldis peale absoluuttermide sisaldada veel ka suhttermide paare (või koosnedagi ainult nendest), kusjuures sama paari termid peavad kuuluma lähteprogrammi ühesse ja samasse sektsiooni ning olema vas-

tandmärgilised (selliste paaride väärtus ei sõltu programmi ümberpaigutamisest mälus).

Suhtavaldised, mis omakorda jagunevad liht- ja liitavaldisteks, võivad peale suhttermide sisaldada ka suvalise arvu absoluutterme.

Lihtavaldised võivad sisaldada ainult paaritu arvu suhtterme, mis kõik peale ühe peavad olema eespoolnimetatud tingimusi rahuldavad paarid, ainsale paaritule (milleks võib muidugi olla ka välisetikett) peab aga tingimata eelnema plussmärk.

Liitavaldised peavad tingimata sisaldama kas ühe negatiivse paaritu suhttermi või siis mitu positiivset ja/või negatiivset paaritut suhttermi (kõikideks avaldisse kuuluvateks paarituteks suhttermideks võivad sealjuures olla ka välisetiketid), kusjuures suhttermide paaride või absoluuttermide olemasolu pole kohustuslik. Liitavaldisi on lubatud kasutada ainult A- ja Y-tüüpi aadresskonstantide (vt. lk. 37) väljendamiseks.

Kui näiteks eeldada, et A ja B on sama sektsiooni suhtetiketid, Y aga mingi välisetikett, siis avaldistest

```
*-A
2*(A-B+5)
L'A+B-A
A-13
=F'11ØØ'
Y-A+*
1Ø-A
A-B-Y
```


7

kolm esimest on absoluutavaldised, kolm järgmist lihtavaldised ja viimased liitavaldised.

Kõikide SS-formaadiga käskdirektiividele (vt. lk. 52) vastavates arvutikäskudes esineb aadressi(de)le lisaks veel ka kas ainult esimese või siis mõlema tehtest osa võtva mäluvälja pikkus. Seejuures võib programmeerija välja pikkuse vastavas direktiivis määrata kas otseselt (näiteks sulgudesse võetud absoluutavaldisena vahetult aadressi määrava avaldise järel) või kaudselt. Esimesel juhul võtab translaator käsu moodustamisel pikkuse võrdseks vastava absoluutavaldise, teisel juhul aga aadressi määrava avaldise pikkusatribuudi ühe võrra vähendatud (seda nõuab formaat!) väärtusega. Viimasel juhul tuleb alati arvestada avaldise pikkusatribuudi määramise ülalnimetatud eeskirju.

Nii näiteks on direktiividest (näites esinevate direktiivide sisulist tähendust selgitatakse hiljem)

	MVC	B(L'C),C
	MVC	A+1Ø,C
	MVC	*+6,=C'TRY␣AK'
	...	. . . . .
A	DS	CL1Ø
	DS	CL2Ø
B	DS	CL3Ø
C	DC	C'PIKKUSATRIBUUT'

esimeses välja pikkus otseselt näidatud, teises ning kolmandas aga kaudselt. Moodustatavatesse käskudesse kannab translaator pikkuse väärtuseks vastavalt 13, 9 ning 5.

4. Konstantide määramine

Mitmesuguste konstantide määramiseks sisaldab ASSEMBLER spetsiaalse direktiivi koodiga DC (Määrata konstant), mille ette kirjutatav etikett tähistab kas seda konstanti või siis ühe direktiiviga mitme konstandi määramisel esimest nendest.

Direktiivi DC aadressosa peab kindlasti sisaldama vähemalt konstandi tunnuse ja konstandi enda (teine ja kuues veerg leheküljel 29 toodud tabelis), kuid neile võib tarbe korral lisada veel järgmisi komponente.

Konstandi tunnusele võib arvkonstantide korral eelneda kordaja, nii arv- kui ka aadresskonstantide korral aga järgneda nn. pikkuse modifikaator. Fiks- ja ujukoma arvude määramisel võib pikkuse modifikaatorile (selle puudumisel konstandi tunnusele) järgneda veel kas mastaabi- või järgumodifikaator või ka mõlemad. Viimasena kirjutatakse aadressossa alati konstant (või mitu konstanti komadega eraldatuna), kusjuures arvkonstandid paigutatakse apostroofide, aadresskonstandid aga sulgude vahele. Tunnuste C, X ja B korral tohib aadressossa kirjutada vaid ühe konstandi.

Konstandi tunnuse ette kirjutatav kordaja näitab, mitu (aadressosa järgnevate komponentidega määratud) konstanti peab translaator mällu paigutama. Tavaliselt esitatakse kordaja kümnendtermi abil, kuid seda võib teha ka sulgudessevõetud absoluutavaldise kaudu (sellesse avaldisse kuuluvad etiketid peavad olema programmis määratud kindlasti enne esinemist avaldises). Kordaja väärtusega üks võib jätta kirjutamata, väärtus null aga omab eritähenduse. Tema korral

nimelt konstanti mällu ei kirjutata ega reserveerita ka mäluvälja, vaid toimub ainult kohaloenduri suurendamine vastava nn. rajastamispiirini (vt. seitsmes veerg leheküljel 29), näiteks poolsõna aadress peab jaguma kahega ning seetõttu toimub rajastamine poolsõnale.

Pikkuse modifikaatori abil näidatakse ühe konstandi paigutamise jaoks nõutava välja pikkus baitides. Tema kirjutamist tuleb alustada tunnusega L, millele vahetult järgneva baitide arvu võib näidata kas konstanttermi või sulgudesse võetud positiivse absoluutavaldisena (mille etiketid peavad eelnevalt määratud olema). Pikkuse modifikaatori kirjutamisel tuleb arvestada lk. 29 kolmandas veerus näidatud piire. Sama tabeli kaheksandas veerus on esitatud pikkuse modifikaatori puudumisel võetavad konstantide standardpikkused (kirjutis "=konst." tähendab seal, et pikkuse määrab direktiivi kirjutatud konstandi sümbolite arv).

Kui direktiivi DC aadressosas esitatav konstant ei sobi talle otseselt või kaudselt määratava pikkusega, siis toimub konstandi kohaldamine mäluväljaga - liigsete sümbolite ärajätmine või siis puuduvate asendamine nullide või tühikutega. Leheküljel 29 toodud tabeli viimases veerus on iga tüübi jaoks näidatud, kummalt poolt vastavat konstanti vajaduse korral kohaldatakse.

Mastaabi modifikaator määrab translaatori tegevuse konstandi moodustamisel. Et see tegevus on fiks- ja ujukoma arvude korral erinev, siis vaatleme seda allpool, seoses erinevate konstantide moodustamise ning mällu paigutamise käsitlemisega. Mastaabi modifikaatori kirjutamist alustatakse

Konstantide määramine (DC)

Konstandi tüüp	Tunnus	Modifikaatorid			Konstant	Rajastamispiir	Kaudne pikkus	Kohaldatakse
		Pikkus	Mastaap	Järk				
Tekstikonstant	C	1 ÷ 256	—	—	'sümbol(id)'	suvaline	=konst.	paremalt
16-ndkonstant	X	1 ÷ 256	—	—	'16-ndarv'	suvaline	=konst.	vasakult
Kahendkonstant	B	1 ÷ 256	—	—	'kahendarv'	suvaline	=konst.	vasakult
Fikskomaarv (sõna)	F	1 ÷ 8	-187 ÷ 346	-85 ÷ 75	'kümnendarv(ud)'	sõna	4	vasakult
Fikskomaarv (poolsõna)	H	1 ÷ 8	-187 ÷ 346	-85 ÷ 75	'kümnendarv(ud)'	poolsõna	2	vasakult
Ujukomaarv (sõna)	E	1 ÷ 8	1 ÷ 14	-85 ÷ 75	'kümnendarv(ud)'	sõna	4	paremalt
Ujukomaarv (topeltsõna)	D	1 ÷ 8	1 ÷ 14	-85 ÷ 75	'kümnendarv(ud)'	topeltsõna	8	paremalt
Pakitud 10-ndkonstant	P	1 ÷ 16	—	—	'kümnendarv(ud)'	suvaline	=konst.	vasakult
Pakkimata 10-ndkonstant	Z	1 ÷ 16	—	—	'kümnendarv(ud)'	suvaline	=konst.	vasakult
Siseaadresskonstant	A	1 ÷ 4	—	—	(absoluutavaldis)	sõna	4	vasakult
		3 või 4	—	—	(suhtavaldis)	sõna	4	vasakult
Siseaadresskonstant	Y	1 või 2	—	—	(absoluutavaldis)	poolsõna	2	vasakult
		2	—	—	(suhtavaldis)	poolsõna	2	vasakult
Otsene aadresskonstant (baasregister + nihe)	S	2	—	—	(absoluutavaldis)	poolsõna	2	vasakult
		2	—	—	(suhtavaldis)	poolsõna	2	vasakult
		2	—	—	(abe. av. (abs. av.))	poolsõna	2	vasakult
Välisaadresekonstant	V	3 või 4	—	—	(välisetikett)	sõna	4	vasakult

direktiivis tunnusega S, millele vahetult järgneb (lk. 29 toodud tabeli neljanda veeru nõuetele vastav) väärtus esitatuna samal viisil nagu pikkuse modifikaatorgi.

Järgu modifikaatori abil võib näidata kümne astme, millega konstant tuleb läbi korrutada enne mäluväljale moodustamist. Selle modifikaatori kirjutamist alustatakse tunnusega E, millele vahetult järgneb (lk. 29 toodud tabeli viienda veeru tingimusi rahuldav) aste samal kujul nagu pikkuse modifikaatoris.

Vaatleme nüüd lähemalt erinevate konstanditüüpide kirjutamist, teisendamist ning mällu paigutamist.

Tekstikonstandi kirjutamisel võib kasutada koodi ДКОИ mistahes sümboleid, ainult sümbolid ' ja & tuleb esitada kahekordselt. Konstant võib sisaldada kuni 256 sümbolit, millest igaüks teisendatakse kaheksakohalisse kahendkoodi ning paigutatakse mälus ühte baiti, kusjuures konstandi alla mineva välja täitmist alustatakse vasakult.

Näiteks direktiivide

C1	DC	C'A&&'
C2	DC	2CL2'AB3'
C3	DC	(L'C1+1)CL3'A1'

alusel paigutab translaator (etiketiga C1 määratud baidist alates) mällu järgmise jada (kokku 15 baiti ehk 3C kuueteistkümnendnumbrit):

C1 5Ø C1 C2 C1 C2 C1 F1 4Ø C1 F1 4Ø C1 F1 4Ø

Etiketi C1 pikkusatribuudiks on siinjuures 2 (määratud kaud-

selt, s.t. konstandi tegeliku pikkusega), etiketil C2 samuti 2 (määratud otseselt pikkuse modifikaatoriga, konstant ise hõivab kordaja tõttu 4 baiti), etiketil C3 aga 3 (määratud otseselt, konstant hõivab kokku 9 baiti).

Kuueteistkümnendkonstanti tohib kirjutada kuni 512 kuueteistkümnendnumbrit, mis translaatori poolt igaüks asendatakse vastava neljakohalise kahendarvuga. Ühte baiti saab seega paigutada kaks kuueteistkümnendnumbrit, kusjuures vastava mäluvälja täitmist alustatakse paremalt. Nii näiteks kannab translaator direktiivide

X1	DC	X'1FA3E'
X2	DC	(L'X1-1)XL1'EF2'
X3	DC	XL(L'X2+2)'18A'

alusel mälu järjestikustesse baitidesse (etiketiga X1 määratud baidist alates) järgmise kuueteistkümnendnumbrite jada:

Ø1 FA 3E F2 F2 ØØ Ø1 8A ,

kusjuures etikettide X1, X2 ja X3 pikkusatribuutideks saadakse vastavalt 3, 1 ja 3.

Kahendkonstant tohib hõivata kuni 256 baiti (konstandi maksimaalne pikkus on seega 2048 bitti), kusjuures vastava mäluvälja täitmist alustab translaator paremalt. Näiteks direktiivide

B1	DC	B'11Ø1'
B2	DC	BL(L'B1+1)'1ØØ111Ø'
B3	DC	BL1'11Ø111Ø1Ø1Ø'

alusel kantakse mällu bitijada

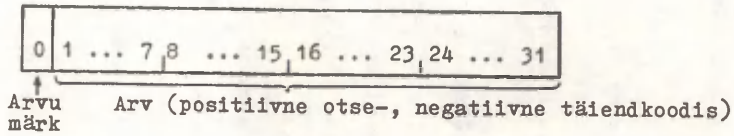
ØØØØ11Ø1 ØØØØØØØØ Ø1ØØ111Ø 111Ø1Ø1Ø ,

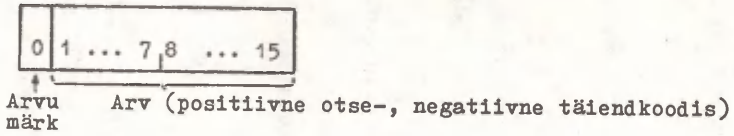
kusjuures L'B1 = 1, L'B2 = 2 ja L'B3 = 1.

Fikskoma arvude kirjutamiseks on kaks võimalust, milledest esimese (tunnus F) korral translaator paigutab konstandi neljabaidisele, teise korral aga (tunnus H) kahebaidisele mäluväljale, kusjuures pikkuse modifikaatori puudumisel toimub kohaloenduri automaatne rajastamine. Fikskoma arve kujutatakse arvutis kahendtäisarvudena: positiivsed arvud on otsekoodis, negatiivsed aga täiendkoodis (vt. lk. 33). Aadressossa võib üksikkonstandi kirjutada täis-, murd- või segakümnendarvuna, kusjuures murdosa eraldatakse punktiga (üldiselt võib punkt esineda kas arvu ees, sees või järel, võib aga ka hoopis puududa). Tarbe korral võib konstant sisaldada veel järgu, mis kirjutatakse vahetult arvu järele kujul En, kus n on märgiga või märgita kümnendarv (jälgida tuleb seda, et arvu järgu ja järgu modifikaatori summa ei ületaks lk. 29 viiendas veerus toodud piire).

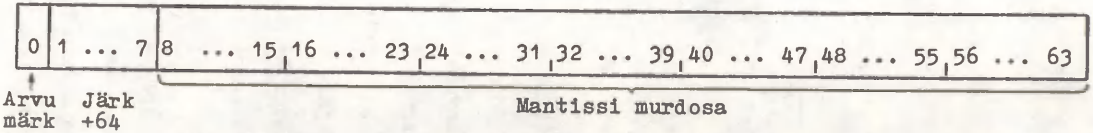
Fikskoma arvude korral võib direktiivi DC aadressosas näidata ka veel mastaabi modifikaatori, mis määrab kahe astme, millega arv tuleb pärast tema kahendkujule teisendamist läbi korrutada. Arvu teisendamine translaatori poolt toimub üldjoontes järgmiselt. Kõigepealt teisendatakse arv kahendsüsteemi, korrutades ta järgumodifikaatori olemasolu korral eelnevalt läbi vastava kümne astmega. Kui teisendamise tulemusena saadakse sega- või murdarv ning mastaabi modifikaator puudub, siis murdosa jäetakse lihtsalt ära, mastaabi modifi-

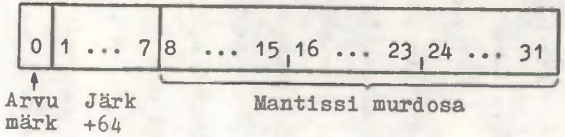
Fikskoma arvude kujutamine arvutis:





Ujukoma arvude kujutamine arvutis:





kaatori esinemise korral aga korrutatakse arv läbi kahe vastava astmega ning ümardatakse. Lõpuks paigutab translaator saadud arvu vastavale mäluväljale, kusjuures negatiivne arv teisendatakse eelnevalt veel täiendkoodi.

Näiteks paigutab translaator direktiivide

F1	DC	F'-128,1ØØ'
F2	DC	2FL3S-3E2'3.563E1'
H	DC	2H'-1'

alusel etiketiga F1 määratud baidist alates mällu kuus fikskoma konstanti, mis kuueteistkümnendnumbritena väljendatuna moodustavad jada:

FF FF FF 8Ø ØØ ØØ ØØ 64 ØØ Ø1 BD ØØ Ø1 BD FF FF FF FF .

Seejuures etikettide F1, F2 ja H pikkusatribuutideks saadakse vastavalt 4, 3 ja 2.

Ka ujukoma arvude kujutamiseks on kaks võimalust. Nendest esimese (tunnus E) korral paigutab translaator konstandi neljabaidilisele, teise korral aga (tunnus D) kaheksabaidilisele mäluväljale (vastavalt sõna ja topeltsõna), kusjuures pikkuse modifikaatori puudumisel toimub kohaloenduri automaatne rajastamine vastavale standardväljale, selle esinemisel aga mitte. Konstandi kirjutamine direktiivi DC aadressossa toimub täpselt samuti nagu fikskoma arvude korral. Erinev on vaid mastaabi modifikaatori tähendus. Nimelt teisendab translaator konstandi kõigepealt kuueteistkümnendsüsteemi normaliseeritud arvuks, seejärel aga nihutab saadud arvu mastaabi modifikaatori olemasolu korral viimase poolt

määratud arvu kuueteistkümnendpositsioonide võrra paremale (vabanevad positsioonid täidetakse nullidega). Alles pärast seda moodustatakse järk (vt. lk. 33) ning konstant paigutatakse vastavale väljale. Kui teisendatud arvu pikkus seejuures ületab välja (otseselt või kaudselt määratud) pikkuse, siis osa mantissi parempoolseid bitte läheb kaduma (vastavalt sellele mantiss ümardatakse). Näiteks direktiivide

E	DC	2E'13.5'
D	DC	DS3E-2'-3675ØE-1'

alusel moodustab translaator etiketiga E määratud baidist alates mällu kolm ujukoma konstanti, mis moodustavad kuueteistkümnendnumbrite jada:

41 D8 ØØ ØØ 41 D8 ØØ ØØ C5 ØØ Ø2 4C ØØ ØØ ØØ ØØ .

Etikettide E ja D pikkusatribuudid saavad vastavalt väärtused 4 ja 8 (näitest parema arusaamise huvides esitame teise direktiiviga määratud konstandi teisendamise käigu:
$-3675ØE-3 = -36750 \cdot 10^{-3} = -36 \cdot 75_{10} = -24.C_{16} = -0.24C_{16} \cdot 16^{2} =$
$= -0.00024C_{16} \cdot 16^{5}$).

Kümnendarve saab arvuti mälus kujutada kahel viisil, seetõttu on ka kümnendkonstantide määramiseks direktiivi DC abil olemas kaks võimalust: tunnus P määrab nn. pakitud kümnendarvu, tunnus Z aga pakkimata e. tsooniformaadis kümnendarvu. Pakkimata kümnendarvude korral sisaldab iga bait ühe kümnendnumbri, s.t. baidi parempoolne poolbait sisaldab numbrile vastava kahendarvu, vasakpoolne aga (nn. tsoon) kahendarvu 1111 (kuueteistkümnendnumbri F).

Erinevus pakkimata kümnendarvu ning infovahetusel kasutatava arvude sümbolesituse vahel seisneb vaid märgi kujutamises. Kui näiteks kaardilt sisestatud või trükiseadmele väljastatava arvu märk kujutatakse (koodis ДКОИ esitatuna, s.t. plussmärgile vastab 4E, miinusmärgile aga 6Ø) arvule vahetult eelnevas baidis (positiivse arvu korral võib hoopiski puududa), siis pakkimata kümnendarvu märk kujutatakse tema poolt hõivatud mäluvälja viimase (parempoolseima) baidi vasakus poolbaidis (F asemel) alati, kusjuures pluss- ja miinusmärki tähistavad vastavalt kuueteistkümnendnumbrid C ja D.

Pakitud kümnendarvude korral, milledega teostatakse näiteks kõik kümnendaritmeetikatehted, sisaldab iga bait kaks kümnendnumbrit, arvu märk esitatakse vastava mäluvälja viimase baidi parempoolses poolbaidis (samuti C ja D abil). Seega kujutab pakkimata kümnendarv endast teatavat vaheastet arvu sümbolesituse ning pakitud kümnendarvu vahel.

Kümnendkonstant kirjutatakse direktiivi DC aadressossa tavalise märgiga või märgita arvuna, mis võib sisaldada ka punkti (arvu teisendamisel jäetakse punkt lihtsalt ära). Nagu nähtub leheküljel 29 toodud tabeli kolmandast veerust ei saa teisendatud kümnendarv hõivata üle kuueteistkümne baidi, mis tähendab, et pakitud kümnendkonstant võib lisaks märgile sisaldada veel kuni kolmkümmend üks, pakkimata aga kuni kuusteist numbrimärki. Kui pikkuse modifikaatorit pole näidatud, siis määrab konstandi poolt hõivatavate baitide arvu tema pikkus pärast teisendamist. Pikkuse modifikaatori esinemisel toimub vajaduse korral kohaldamine. Nii näiteks di-

rektiivide

P1	DC	P'+28.91,-136'
P2	DC	2PL2'12345,-98'
Z	DC	Z'79.76,-589'

alusel saadakse mälus etiketiga P1 määratud baidist alates kaheksa kümnendkonstanti, mis moodustavad järgmise kuueteistkümnendnumbrite jada:

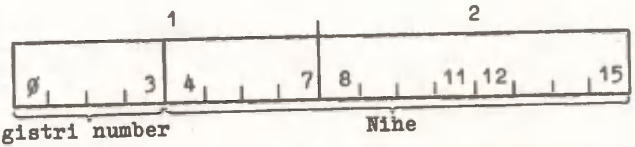
Ø2 89 1C 13 6D 34 5C Ø9 8D 34 5C Ø9 8D F7 F9 F7 C6 F5 F8 D9.

Etikettide P1, P2 ja Z pikkusatribuutideks saadakse vastavalt 3, 2 ja 4 (P1 ja Z korral esimesena kirjutatud konstandi järgi).

Arvkonstantide kõrval saab direktiiviga DC määrata ka aadresskonstante (vt. lk. 29). Aadresskonstant kujutab endast mälu aadressi, mida võib programmis kasutada konstandina, s.t. teostada temaga operatsioone, kanda baas- või indeksregistrisse, muuta või moodustada tema abil käske jne.

Nii A- kui ka Y-tüüpi aadresskonstandi abil on võimalik määrata antud mooduli juurde kuuluva mäluvälja absoluutaadressi, mistõttu neid nimetatakse ka siseaadresskonstantideks. Erinevus nende vahel seisneb vaid moodustatava konstandi pikkuses. Nimelt tohib A-tüüpi konstant hõivata kuni neli baiti (konstanti määrava avaldise väärtus ei tohi seejuures olla suurem kui $2^{31}-1$), Y-tüüpi konstant aga vaid kuni kaks baiti (vastava avaldise väärtus ei tohi olla suurem kui $2^{15}-1$). Mõlema konstanditüübi korral tohivad aga avaldise arvutamisel saadavad vahetulemused hõivata kuni 32 bitti.

S-tüüpi aadresskonstanti saab kasutada otseste aadresside (nihe pluss baasregistri number) määramiseks, kusjuures moodustatav konstant paigutatakse kahte baiti järgmisel viisil:



Märkima peab veel seda, et S-tüüpi aadresskonstanti ei tohi esitada literaalina.

V-tüüpi aadresskonstant võimaldab antud moodulis kasutada mingi teise mooduli juurde kuuluva mäluvälja absoluutaadressi, mistõttu teda nimetatakse ka välisaadresskonstandiks. Seejuures eeldatakse muidugi, et need moodulid komplekteeritakse ühte probleemprogrammi. V-tüüpi aadresskonstandi transleerimisel reserveeritakse neljabaidiline mäluväli ning täidetakse see nullidega, vastav etikett aga kantakse välisetikettide tabelisse. Vajalik absoluutaadress kirjutatakse reserveeritud mäluväljale alles komplekteerimise käigus. Siinkohal tuleks märkida veel seda, et kui V-tüüpi aadresskonstandis esinevat välisetiketti kasutatakse veel antud mooduli muudeski direktiivides, siis tuleb teda täiendavalt kirjeldada direktiivi EXTRN (vt. lk. 45) abil.

Tuleb veel märkida, et kõiki konstante (välja arvatud S-tüüpi aadresskonstant) võib direktiivide aadressosas esitada literaalidena, kirjutades vaid vastava konstandi ette literaali tunnusena sümboli = . Seejuures tohib kordajana ning pikkuse modifikaatorina kasutada ainult märgita, mastaabija järgumodifikaatorina aga ka märgiga kümnendtermi.

5. Mälu jaotamine

Mälu reserveerimise direktiivi DS aadressosa sisaldab täpselt samu komponente, nagu konstantide määramise direktiivi DC aadressosagi, ainult konstandi enese kirjutamine aadressossa pole üldiselt kohustuslik, sest mällu teda ju ei kanta. Kui aga konstant esineb, siis pikkuse modifikaatori puudumisel kasutab translaator seda reserveeritava välja pikkuse määramiseks. Reserveeritud välja(de) edaspidise kasutamise huvides tuleb direktiiv DS enamasti märgendada, kusjuures vastava etiketi pikkusatribuudiks võtab translaator esimese (või ainukese) selle direktiiviga määratud välja pikkuse. Seega konstandi tunnusele eelnev kordaja ei mõjuta pikkusatribuuti, kui kordaja on aga null, siis toimub ainult rajastamine standardpiirini (mälu ei reserveerita). Tuleb veel arvestada, et leheküljel 29 tüüpide C ja X jaoks antud pikkuse maksimaalväärtus direktiivi DS korral ei kehti - ainsaks piiriks on nüüd operatiivmälu maht.

Teatavasti nõuab arvuti adresseerimissüsteem (baas pluss nihe) baasregistri olemasolu. Käskdirektiivi aadressossa kirjutatava aadressi võib seejuures esitada kas otsese aadressina (näidates eraldi nihke ja vastava baasregistri numbri) või siis kaudse aadressina, milleks üldiselt on mingi avaldis (lihtsaimal juhul etikett). Esimesel juhul kasutab translaator arvuti käsu vastavate komponentide moodustamiseks direktiivi aadressosas näidatud avaldiste (tavaliselt kümnendtermid) väärtusi, teisel juhul aga vastavale etiketile varem omistatud väärtust ning programmeerija poolt lähte-

mooduli alguses näidatud baasregistri numbrit. Peale baasregistri numbri näitamise peab programmeerija aga hoolitsema veel selle eest, et programmi täitmise ajaks see register ka sisaldaks vastava baasaadressi. Seetõttu tulebki iga ASSEMBLER-programmi algusesse (enne baseerimist vajavate käskdirektiivide täitmist) kirjutada direktiividepaar

	BALR	B,Ø
	USING	*,B

,

kus B tähistab programmi esimese (või ainukese) baasregistri numbrit määravat absoluutavaldist.

Direktiivi BALR (lähemalt vt. lk. 68) toimel kantakse baasregistrisse järgmise käsu aadress, direktiivi USING (Määrata baasregister) abil aga teatatakse translaatorile baasregister ning näidatakse, et programmi täitmise ajal sisaldab see (baasaadressina) eespoolnimetatud aadressi. Kui aga programm võtab mälus enda alla rohkem kui ühe mälubloki (blokk sisaldab 4096 baiti), siis tuleb direktiivi USING abil näidata vastavalt rohkem baasregistreid. Seejuures translaator eeldab, et teine baasregister sisaldab baasaadressina väärtuse * + 4096, kolmas * + 8192 jne. Vastavate baasaadresside kandmise baasregistritesse peab aga organiseerima programmeerija. Nii näiteks teatatakse direktiive

	BALR	11,Ø
	USING	*,11,4,5,6
A	LM	4,6,BA
	...	. . . . .
BA	DC	A(A+4Ø96,A+8192,A+12288)

sisaldava programmilõiguga translaatorile, et peale registri 11 on baasregistriteks veel registrid 4, 5 ja 6, direktiivi DC abil määratud baasaadressid kantakse aga vastavatesse registritesse käskdirektiivi LM (vt. lk. 54) abil.

Üldjuhul kujutab direktiivi USING esimene operand endast kas lihtavaldist (literaal pole seejuures lubatud) või siis absoluutavaldist, mille väärtus näitab esimeses baasregistris programmi täitmise ajal sisalduvat baasaadressi. Järgmiste operandidena esinevate absoluutavaldiste abil määratakse aga (tarbe korral) kuni kuueteistkümne baasregistri numbrid. Registri number null võib näidata ainult esimese baasregistrina. Seejuures eeldab translaator, et sõltumata esimese operandina näidatud avaldise väärtusest sisaldab esimene baasregister baasaadressina nulli, järgmised baasregistrid aga (kui nad on näidatud) vastavalt väärtusi 4096, 8192 jne.

Baasregistri või selle sisu muutmistest võib programmeerija vajaduse korral teatada direktiivi USING abil programmi mistahes kohas. Nii näiteks teatab direktiividest

	USING	B,12
	...	. . . . .
	USING	B+11ØØ,12

esimene, et baasregister 12 sisaldab baasaadressina väärtuse B, teine aga, et väärtuse B+1100.

Direktiivi USING abil varem määratud baasregistreid saab tarbe korral (näiteks muuks otstarbeks kasutamiseks) tühis-

tada direktiiviga DROP (Tühistada baasregister), mille aadressosas komadega eraldatud absoluutavaldiste kaudu saab näidata kuni kuueteistkümne registri numbrid.

Korraga transleerimisele kuuluvat lähtemoodulit saab vajaduse korral jaotada sektsioonideks, kusjuures eraldi transleeritud lähtemoodulite üksikuid sektsioone on hiljem tarbe korral võimalik ühte programmfaasi komplekteerida. Sektsioonid jagunevad juhtsektsioonideks ning fiktiivseteks sektsioonideks. Juhtsektsioon kujutab endast teatavatele vormistamisnõuetele vastavat iseseisvat programmiosa, fiktiivse sektsiooni abil on aga võimalik kirjeldada mälupiirkondi ilma mälu tegeliku reserveerimiseta (eeldatakse, et mälu reserveeritakse kas mingis selle lähtemooduli osas või siis hoopis teises lähtemoodulis ning seostatakse alles programmi täitmise ajal fiktiivse sektsiooniga).

Lihtsaima struktuuriga lähtemoodul koosneb alati vähemalt ühest juhtsektsioonist, kusjuures esimesele (või ainsale) juhtsektsioonile nime omistamiseks ja kohaloendurile algväärtuse andmiseks võib kasutada direktiivi START (Määrata programmi algus), mille märgend määrab sektsiooni nime (märgendi puudumisel moodustatakse nimeta sektsioon), aadressosas näidatud konstanttermi (oktaabel) väärtus võetakse aga kohaloenduri algväärtuseks (aadressosa puudumisel null). Direktiivi START puudumisel moodustatakse nimeta sektsioon ning kohaloenduri algväärtuseks võetakse null.

Kõik järgnevad direktiivid kuni erinevat märgendit omava direktiivini CSECT (Määrata juhtsektsiooni algus) või direktiivini DSECT (Määrata fiktiivne sektsioon), nende puudumi-

sel aga kuni lähtemooduli lõpuni loetakse esimesse juhtsektsiooni kuuluvaks. Äsjanimetatud direktiivide abil saab kas alustada uut või siis jätkata (varemalustatud samanimelist) juht- või fiktiivset sektsiooni, kusjuures mõlemal direktiivil jääb aadressosa täitmata. Direktiivil CSECT võib puududa ka märgend (ta alustab või jätkab siis nimeta juhtsektsiooni), direktiivil DSECT peab märgend aga tingimata olema.

Samanimeliste sektsioonide eri osad võivad lähtemoodulis paikneda läbisegi, transleerimisel loetakse nad aga terviklikuks, sest translaator formeerib iga sektsiooni jaoks eraldi kohaloenduri. Seejuures iga järgmise juhtsektsiooni kohaloenduri algväärtuseks võetakse eelmisele juhtsektsioonile järgneva esimese topeltsõna aadress, fiktiivse sektsiooni kohaloendurit alustatakse alati nullist (temas esinevatele etikettidele omistatavad väärtused kujutavad endast nihet sektsiooni alguse suhtes). Fiktiivse sektsiooni etikette võib kasutada direktiivide aadressosades, kuid A-tüüpi aadresskonstantides ainult koos teise, vastandmärki omava etiketiga samast sektsioonist.

Fiktiivse sektsiooni seostamiseks mingi mälupiirkonnaga tuleb vastavas juhtsektsioonis direktiivi USING abil teatada baasregister ning näidata baasaadressina mingi etikett sellest fiktiivsest sektsioonist (harilikult näidatakse fiktiivse sektsiooni nimi). Seejärel tuleb aga hoolitseda selle eest, et programmi täitmise ajal oleks vastavas baasregistris selle mäluvälja absoluutaadress, mille struktuuri fiktiivse sektsiooni abil kirjeldati. Tuleb veel arvestada,

et kuigi fiktiivne sektsioon võib sisaldada suvalisi direktiive (mis trükitakse ka transleerimisprotokollis), ei satu ükski nendest objektmoodulisse.

Märgendamata ning aadressosata direktiivi COM (Määrata ühispiirkond) abil saab määrata nn. ühispiirkonna, mis kujutab endast nimeta mälupiirkonda, mille poole võib pöörduda mistahes (üheks programmiks komplekteeritud) sõltumatust moodulist, kus ühispiirkond on määratud. Kui lähtemoodulis esineb mitu direktiivi COM, siis esimene nendest alustab, järgmised aga jätkavad ühispiirkonda. Ühispiirkonna võib mistahes direktiivide abil väljadeks jaotada, kuid transleerimisel objektmoodulisse käske ega konstante ei moodustata (transleerimisprotokollis nad esinevad). Translaator võtab ühispiirkonna kohaloenduri algväärtuseks alati nulli. Kui erinevates moodulites kirjeldatud ühispiirkonnad on erineva pikkusega, siis toimub mälu reserveerimine alati pikima ühispiirkonna jaoks.

Ühispiirkonnas paiknevate mäluväljade kasutamiseks tuleb kõigepealt mingi aadress (harilikult ühispiirkonna algusaadress) kanda baasaadressina mingisse üldregistrisse, see register omakorda aga määrata direktiivi USING abil baasregistriks. Nii näiteks toimub direktiividest

	L	1Ø,=A(ALGUS)
	USING	ALGUS,1Ø
	MVC	A(12),=C'YHISPIIRKOND'
	...	. . . . .
	COM	
ALGUS	DS	1ØF
A	DS	12C

kahe esimese abil baasaadressi (ALGUS) ning baasregistri (1Ø) määramine, kolmanda direktiivi aadressosas võib aga juba kasutada ühispiirkonnas määratud etiketti A.

Eraldi transleeritud, kuid hiljem kokku komplekteeritud moodulite vahelist sidet võimaldavad luua direktiivid ENTRY (Määrata sisendetikett) ja EXTRN (Määrata välisetikett), mille aadressosas võib näidata ühe või (komadega eraldatuna) mitu suhtetiketti (nii sisend- kui ka välisetikette võib üks moodul sisaldada kuni sada). Translaator paigutab andmed nende direktiividega määratud etikettide kohta välisetikettide tabelisse, mille alusel komplekteerimise käigus luuakse juba seosed absoluutaadresside tasemel.

Direktiivi ENTRY kasutatakse nende etikettide määramiseks, mis esinevad küll antud moodulisse kuuluvate programmielementide (käsud, konstandid, reserveeritud mäluväljad) tähistustena, kuid mida kasutatakse ka mingis teises moodulis nendesamade elementidega opereerimiseks (olles seega selle teise mooduli jaoks sisendetikettideks). Nendeks etikettideks ei tohi seejuures olla ei ühispiirkonnas, fiktiivses sektsioonis ega ka nimeta juhtsektsioonis määratud etiketid. Antud mooduli (juhtsektsiooni) nime, mida kasutatakse mingis teises moodulis, pole tarvis direktiivi ENTRY abil kirjeldada, sest translaator paigutab selle niigi alati välisetikettide tabelisse.

Mingis moodulis direktiivi ENTRY abil sisendetikettideks määratud etiketid tuleb neid kasutavas moodulis direktiivi EXTRN abil kirjeldada välisetikettidena. Seejuures tuleb välisetikettidena kirjeldada ka antud moodulis kasutatavate

välismoodulite nimed. Ühtki direktiivi EXTRN aadressosas esinevatest etikettidest ei tohi antud moodulis kasutada märgendina.

Direktiivi EQU (Võtta ekvivalentseks) kasutatakse tema märgendiks oleva etiketi määramiseks aadressosas antud suht- või absoluutavaldise kaudu (avaldises esinevad etiketid peavad tingimata olema määratud eespool). Direktiivi toimel omistab translaator etiketile antud avaldise väärtuse (negatiivse väärtuse korral täiendkoodi 24 madalamat bitti), etiketi pikkusatribuudiks aga võtab avaldise esimese (vasakpoolseima) termi pikkusatribuudi (kui selleks termiks on * või konstant, siis pikkusatribuudiks 1). Direktiiv EQU võimaldab etiketiga tähistada registrinumbreid, vahetuid operande jne., aga samuti ka pikemaid korduvalt tarvisminevaid avaldisi.

Nii näiteks saab programmilõiku

	MVC	A+L'B-1Ø(1Ø),C
	L	7,A+L'B-1Ø
	...	. . . . .
	ST	8,A+L'B-1Ø

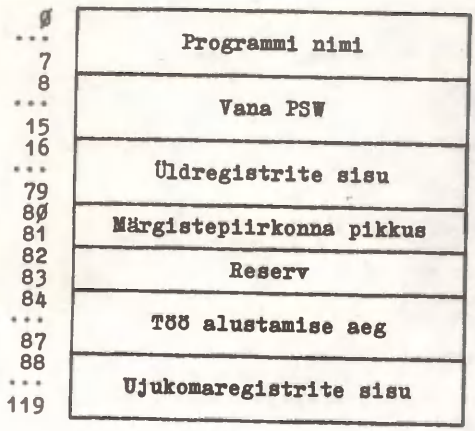
veidi lihtsustada, kui omistada avaldise A+L'B-1Ø väärtus direktiivi EQU abil mingile etiketile:

	MVC	D(1Ø),C
	L	7,D
	...	. . . . .
	ST	8,D
	...	. . . . .
D	EQU	A+L'B+1Ø

6. Programmi seisundi sõna

Operatsioonisüsteemi DOS juhtimisel saab arvuti täita kuni kolme üheaegselt sisemälus paiknevat programmi. Programmidevaheliste üleminekute organiseerimiseks tuleb katkestamise momendil alati fikseerida nii katkestatava programmi kui ka arvuti seisund, et neid töö jätkamisel oleks hiljem võimalik taastada.

Mingi programmi töö katkestamisel saab juhtimise SUPERVIISOR, salvestades kõigepealt selle programmi jätkamiseks vajalikud andmed nn. jao säilituspiirkonda, mis tausta korral asub juhtprogrammi piirkonnas, frontide korral aga vastava jao alguses. Jao säilituspiirkonna 120 baiti täidetakse järgmiselt:



Programmi seisundit iseloomustav informatsioon moodustab nn. programmi seisundi sõna PSW (= Program Status Word), mis programmi täitmise ajal paikneb arvuti juhtseadme eriregist-

rites, katkestamisel aga salvestatakse topeltsõnana järgmise skeemi kohaselt:

0	1	2	3
0 . . . 7	8 . . . 15	16 . . . 23	24 . . . 31
Süsteemne informatsioon		Katkestuskood IC	

4			5	6	7
32.	.35	36...39	40 . . . 47	48 . . . 55	56 . . . 63
IL	CC	PM	Järgmise käsu aadress		

PSW teise poolsõna sisuks olev katkestuskood IC (=Interruption Code) iseloomustab toimunud katkestuse põhjust. Programmeerija seisukohalt olulisemad IC väärtused on järgmised:

0000 - arvuti tõrge;

0001 - vale tehte kood;

0002 - privilegeeritud käsk;

0003 - tõrge käsu EX täitmisel;

0004 - mälukaitse tõrge;

0005 - aadress on lubatust suurem;

0006 - vale spetsifikatsioon;

0007 - viga kümnendandmetes;

0008 - fikskoma ületäitumine;

0009 - viga fikskoma arvude jagamisel;

000A - kümnendületäitumine;

000B - viga kümnendarvude jagamisel;

000C - järgu ületäitumine;

000D - järgu alatäitumine;

ØØØE - väärtuse alatäitumine;

ØØØF - viga ujukoma arvude jagamisel;

ØØ8Ø - katkestus taimerilt;

ØØ4Ø - katkestus kirjutusmasinalt;

ØØKN - katkestus kanali K (= Ø,1,...,6) välisseadmelt number N.

Nendest põhjustest vajavad lähemat selgitust nähtavasti vaid IC väärtused ØØØ4 ja ØØØ6.

Mälukaitse organiseerimiseks on arvuti sisemälu jaotatud 2048-baidisteks blokkideks, mis varustatakse neljabitilise võtmega vastavalt sellele, millise jao juurde antud blokk kuulub. Mingi programmi täitmise ajaks fikseeritakse juhtseadme eriregistris vastava jao kaitsevõti ning iga absoluutaadressi moodustamisel kontrollitakse sellele vastava mälubloki võtit jao kaitsevõtmega, kusjuures nende erinevuse korral toimubki katkestus.

Katkestus vale spetsifikatsiooni alusel toimub järgmistel asjaoludel: operandi absoluutaadress ei jagu vastavalt käsu formaadile kas kahe, nelja või kaheksaga; registritepaare kasutavates käskudes (jagamine, korrutamine, nihutamine jne.) on näidatud paarituarvulise numbriga register; jagaja- või korrutajavälja pikkus kümnendaritmeetikakäskudes on suurem kui kaheksa baiti; tulemusvälja pikkus kümnendarvude korrutamisel või jagamisel on võrdne või väiksem korrutaja- või jagajavälja pikkusest; ujukoma registri number pole lubatav (Ø,2,4,6).

PSW viienda baidi teine pool sisaldab nn. programmi maski PM (= Program Mask), mille abil programmeerija saab nelja

katkestuspõhjuse jaoks katkestamist soovi korral blokeerida. Nimelt on PM neli bitti (PSW bitid 36 - 39) seatud vastavusse järgmiste katkestuspõhjustega:

36 - fikskoma ületäitumine (IC = ØØØ8);

37 - kümnendületäitumine (IC = ØØØA);

38 - järgu alatäitumine (IC = ØØØD);

39 - väärtuse alatäitumine (IC = ØØØE).

PM mingi biti väärtus 1 lubab, väärtus Ø aga keelab katkestuse vastaval põhjusel.

PSW kolm viimast baiti annavad järgmisena täidetava käsu aadressi ja viienda baidi kaks esimest bitti IL (= Instruction Length) selle käsu pikkuse (Ø1 = 2 baiti, 1Ø = 4 baiti, 11 = 6 baiti).

PSW viienda baidi kolmas ja neljas bit sisaldavad tingimuskoodi CC (= Condition Code), mis iseloomustab viimasena täidetud käskdirektiivi (kas vastava operatsiooni tulemuse T, operandide O_1 ja O_2 vahelise suhte, või tekkinud ületäitumise φ järgi). Et enamikus käskdirektiivides leiab kasutamist vaid üks viiest standardsest CC moodustamise eeskirjast, siis lühiduseks tähistame neid eeskirju edaspidi järgmiselt:

Eeskirja tähis	CC väärtus			
	ØØ	Ø1	1Ø	11
a	$T = Ø$	$T < Ø$	$T > Ø$	φ
b	$T = Ø$	$T \neq Ø$	-	-
c	$T = Ø$ ülekandeta	$T \neq Ø$ ülekandeta	$T = Ø$ ülekandega	$T \neq Ø$ ülekandega
d	$O_1 = O_2$	$O_1 < O_2$	$O_1 > O_2$	-
-	säilib eelmine CC			

7. Käskdirektiivid

Käesolevas punktis vaadeldavad direktiivid vastavad enam-vähem üksüheselt arvuti käskudele ja neid on seetõttu loomulik nimetada käskdirektiivideks. Parema ülevaate saamiseks on kõik käskdirektiivid rühmitatud vastavalt nende funktsioonidele (salvestamine, aritmeetika, loogika jne.) ning koondatud lehekülgedel 59-71 toodud tabelitesse.

Nende tabelite neljandas veerus näidatakse iga direktiivi jaoks tema formaat, mis määrab aadressosa esitamise viisi. Formaatide tähistena kasutatakse vastavate ingliskeelsete nimetuste lühendeid (RR = Register-Register; RX = Register--indeXed storage; RS = Register-Storage; SI = Storage-Immediate operand; SS = Storage-Storage). Formaaditähiste selgitus on toodud leheküljel 52, kus iga formaadi korral näidatakse nii direktiivi aadressosa kuju kui ka vastava käsu esitus mälus. Direktiivide aadressosade kujutamisel on kasutatud järgmisi tähistusi:

R - registri numbrina tõlgendatav absoluutavaldis (kahe registriga formaatides R_1 ja R_2);

X - indeksregistri numbrina tõlgendatav absoluutavaldis;

I - vahetu operandina tõlgendatav absoluutavaldis;

M - arv vahemikust Ø - 15;

L - kuni 256-baidise välja pikkusena tõlgendatav absoluutavaldis;

L_1,L_2 - kuni 16-baidiste väljade pikkustena tõlgendatavad absoluutavaldised;

A - mälu aadressina tõlgendatav konstruktsioon, mis tu-

Direktiivide formaadid

Formaadi		Käsu struktuur mälus (baidid)												Direktiivi aadressosa
nimetus	tähis	1	2		3		4		5		6			
register – register	RR	kood	R_1	R_2										R_1, R_2
	RR'	kood	R											R
	RR"	kood	I											I
	RR"'	kood	M	R										M,R
register – indekseeritud mälu	RX	kood	R	X	B		D							R,A(X)
	RX'	kood	M	X	B		D							M,A(X)
register – mälu	RS	kood	R_1	R_2	B		D							R_1, R_2,A
	RS'	kood	R		B		D							R,A
mälu – vahetu operand	SI	kood	I		B		D							A,I
mälu – mälu	SS	kood	L_1-1	L_2-1	B_1		D_1		B_2		D_2			$A_1(L_1), A_2(L_2)$
	SS'	kood	L-1		B_1		D_1		B_2		D_2			$A_1(L), A_2$

leb direktiivis esitada kas ühe (absoluut- või suht-) aval-disena S, või kujul D(B), kus absoluutavaldised D ja B määravad vastavalt nihke ja baasi (kahe aadressiga formaatides A_1 ja A_2);

A(X) - indekseeritud aadressina tõlgendatav konstruktsioon, mis tuleb esitada kas kujul S(X) või D(X,B);

A(L) - pikkusega varustatud aadressina tõlgendatav konstruktsioon, mis tuleb esitada kas kujul S(L) või D(L,B).

Lehekülgedel 59-71 toodud tabelites tähistatakse aadresse esitusviisist sõltumata ikka tähega A (kahe aadressiga direktiivides A_1 ja A_2). Ühtlasi tehakse seal vahet üldregistri numbri R (või R_1 ja R_2) ning ujukomaregistri numbri $\bar{R}$ (või $\bar{R}_1$ ja $\bar{R}_2$) vahel. Sulud tehte kirjelduses tähendavad vastava mäluvälja või registri sisu, kusjuures ülemise indeksiga näidatakse välja pikkust baitides. Näiteks (A^L) tähendab L-baidilise välja sisu, kus esimese baidi aadressiks on A; (R^1) tähendab üldregistri R parempoolseima baidi sisu; $(\bar{R}_2^4)$ tähendab ujukomaregistri $\bar{R}_2$ nelja vasakpoolseima baidi sisu jne. Kui tegemist on terve registriga, siis R^4 ja $\bar{R}^8$ asemel kirjutatakse lihtsalt R ja $\bar{R}$. Kirjutis R,R+1 tähendab kaht järjestikust registrit (kokku 8 baiti), kus R peab olema paarisnumbriga.

Käskdirektiive kirjeldavate tabelite viimases veerus on antud tingimuskoodi CC moodustamise standardeeskirja tähis vastavalt leheküIje 50 tabelile. Kui aga (ebastandardne) eeskiri kirjeldatakse tehte sisu selgitamisel, siis veerus CC märgitakse seda tähisega ⇐ .

Käskdirektiivide sisuline tähendus selgub enamasti vas-

tavast tabelist. Täiendavaid kommentaare vajavad nähtavasti vaid üksikud direktiivid.

Sõnaderühma ülekandmist võimaldavate salvestamisdirektiivide LM ja STM kasutamisel tuleb arvestada, et $R_2 < R_1$ korral toimub tsükliline üleminek registrilt numbriga 15 registrile numbriga Ø. Nii näiteks direktiivi

	LM	14, 2, A

täitmisel kantakse viis sõna etiketiga A määratud baidist alates registritesse numbritega 14, 15, Ø, 1 ja 2.

Aritmeetikadirektiivide korral tuleb arvestada, et terminiga "kood" tähistatakse vastaval väljal paiknevat bitiridа. Katkestus põhjusel "viga jagamisel" tekib arvutüübist sõltuvalt järgmistel asjaoludel. Fikskoma arvude jagamisel (direktiivid D ja DR) tekib katkestus IC = ØØØ9 kas siis, kui jagatis ei mahu registrisse R+1 (resp. R_1+1), või kui jagaja võrdub nulliga. Kümnendarvude jagamisel (direktiiv DF) kutsub katkestuse IC = ØØØR esile kas nulliga jagamise katse või jagatise alla mineva välja pikkuse (L_1-L_2) ebapiisavus. Ujukoma arvude jagamisel (direktiivid DE, DD, DER ja DDR) toimub katkestus IC = ØØØF vaid nulliga jagamise katsel.

Loogilise võrdlemise direktiivide (CL, CLR, CLC ja CLI) täitmisel toimub koodide bitiviisiline võrdlemine vasakult paremale. Tehe lõpetatakse (ja moodustatakse CC) esimese mittevõrdee bitipaari leidumisel.

Nihutamisdirektiivide korral määravad nihutatavate bittide arvu aadressi A kuus madalamat bitti. Topeltsõnade arit-

meetilieel nihutamisel tuleb seejuures silmas pidada veel seda, et arvu märgi määrab esimese (paarisarvulise numbriga) registri esimene bitt.

Suunamisdirektiivid on toodud kahes tabelis, milledest esimene sisaldab n.ö. põhidirektiivid, teine aga tingimusliku suunamisdirektiivi BC (RCR) modifikatsioonid. Viimaste kasutamine lihtsustab programmeerimist, sest jääb ära suunamistingimuse moodustamine (ülevaatlikkuse huvides on lehekülje 69 viimastes lahtrites ära toodud ka igale modifikatsioonile vastav põhidirektiiv koos suunamistingimusega).

Veidi pikemat selgitamist vajavad mõned teisendamisdirektiivid (vt. lk. 70), eeskätt just redigeerimisdirektiivid (ED ja EDMK), mis on ette nähtud pakitud kümnendarvude teisendamiseks väljatrükkimise jaoks sobivale kujule.

Redigeerimine toimub vastava nn. šablooni alusel, mis peab enne direktiivi täitmist asuma aadressiga A_1 määratud väljal. Samale väljale (šablooni asemele) saadakse ka redigeeritud arv. Redigeerimise käigus arvud teisendatakse pakkimata kümnendarvudeks, kusjuures tarbe korral kustutatakse tüvenumbritele eelnevad nullid, lisatakse arvu märk, mõõtühik jne.

Igale redigeeritavale numbrile väljalt A_2 peab šabloonis vastama üks sümbol (bait). Vastavate redigeerimisel saadavate numbribaitide asukoht ning sisu tulemusväljal A_1 sõltub aga allpoolvaadeldavate sümbolite abil määratavast formaadist.

Šablooni esimeseks sümboliks on nn. täitesümbol, millena võib kasutada mistahes trükisümboli kuueteistkümnendkoodi

(sagedamini kasutatakse tühikut koodiga 4Ø).

Väljalt A_2 järjekordse numbri valimiseks ning redigeerimiseks tuleb kasutada sümbolit kuueteistkümnendkoodiga 2Ø, samaks tegevuseks koos eelnevate nullide kustutamise lõpetamisega aga sümbolit kuueteistkümnendkoodiga 21. Need sümbolid asendatakse redigeerimisel kas täitesümboliga või siis vastava numbriga väljalt A_2.

Eraldamissümbolit koodiga 22 tuleb kasutada redigeeritud arvude vahekoha näitamiseks juhul kui ühe direktiiviga redigeeritakse mitu järjestikust arvu väljalt A_2. Redigeerimisel asendatakse eraldamissümbolid täitesümbolitega.

Soovi korral võib redigeeritud arvule lisada mistahes trükisümboleid (nn. teatesümbolid), nagu näiteks mõõtühik, eraldaja vms., näidates šabloonis nendele vastavad kuueteistkümnendkoodid.

Redigeerimisdirektiivi täitmisel toimub šablooni ning redigeeritava arvu paralleelne läbivaatamine vasakult paremale. Seejuures sõltub tehte tulemus spetsiaalsest indikaatorist, mis juhib tüvenumbritele eelnevate nullide kustutamist ja teatesümbolite trükkimist ning mis redigeerimisprotsessi jooksul lülitatakse kas välja (enne redigeerimise algust ning hiljem kas väljal A_2 plussmärgi või šabloonis eraldamissümboli esinemisel) või sisse (kas väljalt A_2 tüvenumbri valimisel - kood šabloonis 2Ø või 21, või siis väljalt A_2 eelnevatest nullidest viimase valimisel - kood šabloonis 21). Väljalülitatud indikaatori korral asendatakse kõik šabloonis esinevad teatesümbolid, aga samuti ka eelnevad nullid väljalt A_2 (millele šabloonis vastab kood 2Ø)

täitesümbolitega; sisselülitatud indikaatori korral jäävad teatesümbolid muutmata, Šablooni koodidele 2Ø ja 21 vastavad numbrid väljalt A_2 teisendatakse aga pakkimata kujule.

Nii näiteks direktiividega

	MVC	V,=X'4Ø2Ø2Ø216Ø222Ø2Ø6B21'
	ED	V,ARVUD
	...	. . . . .
V	DS	CL1Ø
ARVUD	DC	PL2'-13,25'

määratud redigeerimistehte täitmine toimub järgmises tabelis näidatud viisil (s = sees, v = väljas):

Šabloon	4Ø	2Ø	2Ø	21	6Ø	22	2Ø	2Ø	6B	21
Arvud		Ø	1	3	D		Ø	2		5
Indikaator	v	v	s	s	s	v	v	s	s	s
Redigeerimistulemus	4Ø	4Ø	F1	F3	6Ø	4Ø	4Ø	F2	6B	F5
Trükis			1	3	-			2	,	5

Direktiive TR ning TRT kasutatakse lähteandmete ümberkodeerimiseks ja kontrollimiseks ning nad leiavad rakendamist näiteks infovahetuse organiseerimisel perfolindiga.

Eridirektiividest võimaldab direktiiv EX modifitseerida ning täita üksikut, näiteks konstantide seas asuvat käsku,

kusjuures programmi täitmine jätkub direktiivile EX järgnevast direktiivist. Kui täidetav direktiiv on seejuures omakorda EX, siis toimub programmkatkestus koodiga IC = ØØØ3. Direktiivi EX abil saab täitmise ajaks formeerida selliseid käsu teises baidis sisalduvaid komponente nagu registri number, välja pikkus jms. (mälus jääb käsk muutmata).

Näiteks direktiivide

	L	1Ø, = F'5'
	EX	5,K
	...	. . . .
K	MVC	A(Ø),B

toimel formeeritakse välja pikkus (5) aadressiga K+1 määratud baiti ning seejärel täidetakse käsk (MVC) aadressil K.

Direktiivi SVC mitmesugused (aadressosas oleva vahetu operandiga määratavad) modifikatsioonid leiavad kasutamist nii SUPERVIISORi poole pöördumise, kui ka teiste süsteemsete makrokäskude makrolaiendites. Kaks sagedasemat kasutamisviisi on järgmised.

Iga probleemprogrammi viimasena täidetavaks direktiiviks peab olema näiteks SVC 14 , mille toimel juhtimine antakse tagasi SUPERVIISORile.

Mingi faasi paigaldamist C-teegist ning juhtimise andmist temale võimaldab direktiiv SVC 1 . Seejuures peab üldregistritesse 1 ja Ø eelnevalt olema kantud vastavalt faasi nime sisaldava kaheksabaidise mäluvälja ja faasi sisendpunkti absoluutaadressid.

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Kanda sõna	LR	18	RR	$(R_2) \rightarrow R_1$	-
Kanda uk-sõna (ujukoma sõna)	LER	38	RR	$(\bar{R}_2^4) \rightarrow \bar{R}_1^4$	-
Kanda uk-topeltsõna	LDR	28	RR	$(\bar{R}_2) \rightarrow \bar{R}_1$	-
Kanda sõna plussiga	LPR	1Ø	RR	$\lvert(R_2)\rvert \rightarrow R_1$	a
Kanda uk-sõna plussiga	LPER	3Ø	RR	$\lvert(\bar{R}_2^4)\rvert \rightarrow \bar{R}_1^4$	a
Kanda uk-topeltsõna plussiga	LPDR	2Ø	RR	$\lvert(\bar{R}_2)\rvert \rightarrow \bar{R}_1$	a
Kanda sõna miinusega	LNR	11	RR	$-\lvert(R_2)\rvert \rightarrow R_1$	a
Kanda uk-sõna miinusega	LNER	31	RR	$-\lvert(\bar{R}_2^4)\rvert \rightarrow \bar{R}_1^4$	a
Kanda uk-topeltsõna miinusega	LNDR	21	RR	$-\lvert(\bar{R}_2)\rvert \rightarrow \bar{R}_1$	a
Kanda ja kontrollida sõna	LTR	12	RR	$(R_2) \rightarrow R_1$	a
Kanda ja kontrollida uk-sõna	LTER	32	RR	$(\bar{R}_2^4) \rightarrow \bar{R}_1^4$	a
Kanda ja kontrollida uk-topeltsõna	LTDR	22	RR	$(\bar{R}_2) \rightarrow \bar{R}_1$	a

Salvestamisdirektiivid (järg)

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Kanda sõna vastasmärgiga	LCR	13	RR	$-(R_2) \rightarrow R_1$	a
Kanda uk-sõna vastasmärgiga	LCER	33	RR	$-(\bar{R}_2^4) \rightarrow \bar{R}_1^4$	a
Kanda uk-topeltsõna vastasmärgiga	LCDR	23	RR	$-(\bar{R}_2) \rightarrow \bar{R}_1$	a
Tõsta sümbol	MVI	92	SI	$I \rightarrow A^1$	-
Tõsta sümbolid	MVC	D2	SS'	$(A_2^L) \rightarrow A_1^L$	-
Tõsta numbrid	MVN	D1	SS'	$(A_2^L) \rightarrow A_1^L$ parempoolsed poolbaidid	-
Tõsta tsoonid	MVZ	D3	SS'	$(A_2^L) \rightarrow A_1^L$ vasakpoolsed poolbaidid	-
Tõsta nihkega	MVO	F1	SS	$(A_2^{L_2}) \rightarrow A_1^{L_1}$ nihkega 4 bitti vasakule	-
Tõsta pakitud 10-ndarv koos nullimisega	ZAP	F8	SS	$(A_2^{L_2}) \rightarrow A_1^{L_1}$ täitmata jäänud poolbaidid nullitakse	a

Salvestamisdirektiivid (järg)

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Kanda aadress	LA	41	RX	$A \rightarrow R$	-
Kanda sümbol	IC	43	RX	$(A^1) \rightarrow R^1$	-
Kanda poolsõna	LH	48	RX	$(A^2) \rightarrow R^2$	-
Kanda sõna	L	58	RX	$(A^4) \rightarrow R$	-
Kanda uk-sõna	LE	78	RX	$(A^4) \rightarrow \overline{R}^4$	-
Kanda uk-topeltsõna	LD	68	RX	$(A^8) \rightarrow \overline{R}$	-
Kanda korduvalt	LM	98	RS	$(A^4), \ldots \rightarrow R_1, \ldots, R_2$	-
Salvestada sümbol	STC	42	RX	$(R^1) \rightarrow A^1$	-
Salvestada poolsõna	STH	4Ø	RX	$(R^2) \rightarrow A^2$	-
Salvestada sõna	ST	5Ø	RX	$(R) \rightarrow A^4$	-
Salvestada uk-sõna	STE	7Ø	RX	$(\overline{R}^4) \rightarrow A^4$	-
Salvestada uk-topeltsõna	STD	6Ø	RX	$(\overline{R}) \rightarrow A^8$	-
Salvestada korduvalt	STM	9Ø	RS	$(R_1), \ldots, (R_2) \rightarrow A^4, \ldots$	-

Aritmeetikadirektiivid

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Liita	A	5A	RX	(R) + (A^4) → R	a
Liita poolsõnaga	AH	4A	RX	(R) + (A^2) → R	a
Liita registrid	AR	1A	RR	(R_1) + (R_2) → R_1	a
Liita koodid	AL	5E	RX	kood (R) + kood (A^4) → R	c
Liita koodid registrites	ALR	1E	RR	kood (R_1) + kood (R_2) → R_1	c
Lahutada	S	5B	RX	(R) - (A^4) → R	a
Lahutada poolsõna	SH	4B	RX	(R) - (A^2) → R	a
lahutada registrid	SR	1B	RR	(R_1) - (R_2) → R_1	a
lahutada koodid	SL	5F	RX	kood (R) - kood (A^4) → R	c
lahutada koodid registrites	SLR	1F	RR	kood (R_1) - kood (R_2) → R_1	c
Korrutada	M	5C	RX	(R+1) × (A^4) → R,R+1	-
Korrutada poolsõnaga	MH	4C	RX	(R) × (A^2) → R	-
Korrutada registrid	MR	1C	RR	(R_1+1) × (R_2) → R_1,R_1+1	-
Jagada	D	5D	RX	(R,R+1)/(A^4) jääk → R jagatis → R+1	-
Jagada registrid	DR	1D	RR	(R_1,R_1+1)/(R_2) jääk → R_1 jagatis → R_1+1	-

Aritmeetikadirektiivid (järg)

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Liita kümnendarvud	AP	FA	SS	$(A_1^{L_1}) + (A_2^{L_2}) \rightarrow A_1^{L_1}$	a
Lahutada kümnendarvud	SP	FB	SS	$(A_1^{L_1}) - (A_2^{L_2}) \rightarrow A_1^{L_1}$	a
Korrutada kümnendarvud	MP	FC	SS	$(A_1^{L_1}) \times (A_2^{L_2}) \rightarrow A_1^{L_1}$	-
Jagada kümnendarvud	DP	FD	SS	$(A_1^{L_1})/(A_2^{L_2})$ jagatis $\rightarrow A_1^{L_1-L_2}$ jääk $\rightarrow A_1^{+L_1-L_2}\,\vert^{L_2}$	-
Korrutada sõnad	ME	7C	RX	$(\bar{R}^4) \times (A^4) \rightarrow \bar{R}^4$	-
Korrutada topeltsõnad	MD	6C	RX	$(\bar{R}) \times (A^8) \rightarrow \bar{R}$	-
Korrutada sõnad registrites	MER	3C	RR	$(\bar{R}_1^4) \times (\bar{R}_2^4) \rightarrow \bar{R}_1^4$	-
Korrutada topeltsõnad reg.	MDR	2C	RR	$(\bar{R}_1) \times (\bar{R}_2) \rightarrow \bar{R}_1$	-
Jagada sõnad	DE	7D	RX	$(\bar{R}^4) / (A^4) \rightarrow \bar{R}^4$	-
Jagada topeltsõnad	DD	6D	RX	$(\bar{R}) / (A^8) \rightarrow \bar{R}$	-
Jagada sõnad registrites	DER	3D	RR	$(\bar{R}_1^4) / (\bar{R}_2^4) \rightarrow \bar{R}_1^4$	-
Jagada topeltsõnad reg.	DDR	2D	RR	$(\bar{R}_1) / (\bar{R}_2) \rightarrow \bar{R}_1$	-
Poolitada sõna	HER	34	RR	$(\bar{R}_2^4) / 2 \rightarrow \bar{R}_1^4$	-
Poolitada topeltsõna	HDR	24	RR	$(\bar{R}_2) / 2 \rightarrow \bar{R}_1$	-

Aritmeetikadirektiivid (järg)

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Liita sõnad normaliseerimisega	AE	7A	RX	$(\bar{R}^4) + (A^4) \xrightarrow{n} \bar{R}^4$	a
Liita topeltsõnad normaliseerimisega	AD	6A	RX	$(\bar{R}) + (A^8) \xrightarrow{n} \bar{R}$	a
Liita sõnad registrites norm-ga	AER	3A	RR	$(\bar{R}_1^4) + (\bar{R}_2^4) \xrightarrow{n} \bar{R}_1^4$	a
Liita topeltsõnad registrites norm-ga	ADR	2A	RR	$(\bar{R}_1) + (\bar{R}_2) \xrightarrow{n} \bar{R}_1$	a
Liita sõnad normaliseerimiseta	AU	7E	RX	$(\bar{R}^4) + (A^4) \rightarrow \bar{R}^4$	a
Liita topeltsõnad normaliseerimiseta	AW	6E	RX	$(\bar{R}) + (A^8) \rightarrow \bar{R}$	a
Liita sõnad registrites norm-ta	AUR	3E	RR	$(\bar{R}_1^4) + (\bar{R}_2^4) \rightarrow \bar{R}_1^4$	a
Liita topeltsõnad registrites norm-ta	AWR	2E	RR	$(\bar{R}_1) + (\bar{R}_2) \rightarrow \bar{R}_1$	a
Lahutada sõnad normaliseerimisega	SE	7B	RX	$(\bar{R}^4) - (A^4) \xrightarrow{n} \bar{R}^4$	a
Lahutada topeltsõnad normaliseerimisega	SD	6B	RX	$(\bar{R}) - (A^8) \xrightarrow{n} \bar{R}$	a
Lahutada sõnad registrites norm-ga	SER	3B	RR	$(\bar{R}_1^4) - (\bar{R}_2^4) \xrightarrow{n} \bar{R}_1^4$	a
Lahutada topeltsõnad registrites norm-ga	SDR	2B	RR	$(\bar{R}_1) - (\bar{R}_2) \xrightarrow{n} \bar{R}_1$	a
Lahutada sõnad normaliseerimiseta	SU	7F	RX	$(\bar{R}^4) - (A^4) \rightarrow \bar{R}^4$	a
Lahutada topeltsõnad normaliseerimiseta	SW	6F	RX	$(\bar{R}) - (A^8) \rightarrow \bar{R}$	a
Lahutada sõnad registrites norm-ta	SUR	3F	RR	$(\bar{R}_1^4) - (\bar{R}_2^4) \rightarrow \bar{R}_1^4$	a
Lahutada topeltsõnad registrites norm-ta	SWR	2F	RR	$(\bar{R}_1) - (\bar{R}_2) \rightarrow \bar{R}_1$	a

Loogikadirektiivid

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Loogiliselt korrutada sõnaga	N	54	RX	$(R) \wedge (A^4) \rightarrow R$	b
Loogiliselt korrutada registrid	NR	14	RR	$(R_1) \wedge (R_2) \rightarrow R_1$	b
Loogiliselt korrutada vahetu operandiga	NI	94	SI	$(A^1) \wedge I \rightarrow A^1$	b
Loogiliselt korrutada sümbolid	NC	D4	SS'	$(A_1^L) \wedge (A_2^L) \rightarrow A_1^L$	b
Loogiliselt liita sõnaga	O	56	RX	$(R) \vee (A^4) \rightarrow R$	b
Loogiliselt liita registrid	OR	16	RR	$(R_1) \vee (R_2) \rightarrow R_1$	b
Loogiliselt liita vahetu operandiga	OI	96	SI	$(A^1) \vee I \rightarrow A^1$	b
Loogiliselt liita sümbolid	OC	D6	SS'	$(A_1^L) \vee (A_2^L) \rightarrow A_1^L$	b
Mittesamaväärsustada sõnaga	X	57	RX	$(R) \approx (A^4) \rightarrow R$	b
Mittesamaväärsustada registrid	XR	17	RR	$(R_1) \approx (R_2) \rightarrow R_1$	b
Mittesamaväärsustada vahetu operandiga	XI	97	SI	$(A^1) \approx I \rightarrow A^1$	b
Mittesamaväärsustada sümbolid	XC	D7	SS'	$(A_1^L) \approx (A_2^L) \rightarrow A_1^L$	b

Võrdlemisdirektiivid

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Võrrelda sõnad	C	59	RX	$(R) \gtrless (A^4)$?	d
Võrrelda poolsõnad	CH	49	RX	$(R^2) \gtrless (A^2)$?	d
Võrrelda registrid	CH	19	RR	$(R_1) \gtrless (R_2)$?	d
Võrrelda uk-sõnad	CE	79	RX	$(\bar{R}^4) \gtrless (A^4)$?	d
Võrrelda uk-topeltsõnad	CD	69	RX	$(\bar{R}) \gtrless (A^8)$?	d
Võrrelda uk-sõnad registrites	CER	39	RR	$(\bar{R}_1^4) \gtrless (\bar{R}_2^4)$?	d
Võrrelda uk-topeltsõnad registrites	CDR	29	RR	$(\bar{R}_1) \gtrless (\bar{R}_2)$?	d
Loogiliselt võrrelda sõnad	CL	55	RX	$(R) \gtrless (A^4)$?	d
Loogiliselt võrrelda registrid	CLR	15	RR	$(R_1) \gtrless (R_2)$?	d
Loogiliselt võrrelda sümbolid	CLC	D5	SS'	$(A_1^L) \gtrless (A_2^L)$?	d
Loogiliselt võrrelda vahetu operandiga	CLI	95	SI	$(A^1) \gtrless I$?	d
Võrrelda kümnendarvud	CP	F9	SS	$(A_1^{L_1}) \gtrless (A_2^{L_2})$? võrdlemine toimub paremalt vasakule	d

Nihutamisdirektiivid

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Nihutada sõna loogiliselt paremale	SRL	88	RS'	(R) → nihutamisega paremale → R	–
Nihutada topeltsõna loogiliselt paremale	SRDL	8C	RS'	(R,R+1) → nihutamisega paremale → R,R+1	–
Nihutada sõna loogiliselt vasakule	SLL	89	RS'	(R) → nihutamisega vasakule → R	–
Nihutada topeltsõna loogiliselt vasakule	SLDL	8D	RS'	(R,R+1) → nihutamisega vasakule → R,R+1	–
Nihutada sõna aritmeetiliselt paremale	SRA	8A	RS'	(R) → nihutamisega paremale → R	a
Nihutada topeltsõna aritmeetiliselt paremale	SRDA	8E	RS'	(R,R+1) → nihutamisega paremale → R,R+1	a
Nihutada sõna aritmeetiliselt vasakule	SLA	8B	RS'	(R) → nihutamisega vasakule → R	a
Nihutada topeltsõna aritmeetiliselt vasakule	SLDA	8F	RS'	(R,R+1) → nihutamisega vasakule → R,R+1	a

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu		CC
Minna tingimuskoodi järgi	BC	47	RX'	M ∧ CC = CC korral juhtimine aadressile A, muidu aadressile * + 4		-
Minna tingimuskoodi järgi registri kaudu	BCR	Ø7	RR''	M ∧ CC = CC ja R ≠ Ø korral juhtimine aadressile (R), muidu aadressile * + 2		-
Minna loendaja järgi	BCT	46	RX	(R) - 1 → R; (R) ≠ Ø korral juhtimine aadressile A, muidu aadressile * + 4		-
Minna loendaja järgi registri kaudu	BCTR	Ø6	RR	$(R_1) - 1 \rightarrow R_1$; $(R_1) \neq Ø$ ja $R_2 \neq Ø$ korral juhtimine aadressile (R_2), muidu * + 2		-
Minna naasmisega	BAL	45	RX	* + 4 → R; juhtimine aadressile A		-
Minna naasmisega registri kaudu	BALR	Ø5	RR	* + 2 → R_1; $R_2 \neq Ø$ korral juhtimine aadressile (R_2), muidu aadressile * + 2		-
Minna suurema indeksi järgi	BXH	86	RS	$(R_1) + (R_2) \rightarrow R_1$; $(R_1) > (R_2')$ korral juhtimine aadressile A	paaritu R_2' = kas R_2 või $R_2 + 1$	-
Minna väiksem-võrdse indeksi järgi	BXLE	87	RS	$(R_1) + (R_2) \rightarrow R_1$; $(R_1) \leq (R_2')$ korral juhtimine aadressile A		

Suunamisdirektiivi BC (BCR) modifikatsioonid

Tehte nimetus	Modif. kuju	Vastava BC kuju
Üldkasutatavad		
Minna	B A(X)	BC 15,A(X)
Minna registri kaudu	BR R	BCR 15,R
Tühidirektiiv	NOP A(X)	BC Ø,A(X)
Tühidirektiiv	NOPR R	BCR Ø,R
Võrdlemisdirektiivi järel		
Minna, kui $O_1 > O_2$	BH A(X)	BC 2,A(X)
Minna, kui $O_1 \geq O_2$	BNL A(X)	BC 11,A(X)
Minna, kui $O_1 = O_2$	BE A(X)	BC 8,A(X)
Minna, kui $O_1 \leq O_2$	BNH A(X)	BC 13,A(X)
Minna, kui $O_1 < O_2$	BL A(X)	BC 4,A(X)
Minna, kui $O_1 \neq O_2$	BNE A(X)	BC 7,A(X)

Tehte nimetus	Modif. kuju	Vastava BC kuju
Aritmeetikadirektiivi järel		
Minna, kui T > Ø	BP A(X)	BC 2,A(X)
Minna, kui T ≥ Ø	BNM A(X)	BC 11,A(X)
Minna, kui T = Ø	BZ A(X)	BC 8,A(X)
Minna, kui T ≤ Ø	BNP A(X)	BC 13,A(X)
Minna, kui T < Ø	BM A(X)	BC 4,A(X)
Minna, kui T ≠ Ø	BNZ A(X)	BC 7,A(X)
Minna ületäitumisel	BO A(X)	BC 1,A(X)
Direktiivi TM järel		
Minna, kui kõik ühed	BO A(X)	BC 1,A(X)
Minna, kui ühed-nullid	BM A(X)	BC 4,A(X)
Minna, kui kõik nullid	BZ A(X)	BC 8,A(X)

Teisendamisdiraktiivid

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Pakkida	PACK	P2	SS	Pakkimata kümnendarv väljalt $A_2^{L_2}$ teisendatakse pakitud kümnendarvuks väljale $A_1^{L_1}$	-
Lahti pakkida	UNPK	P3	SS	Pakitud kümnendarv väljalt $A_2^{L_2}$ teisendatakse pakkimata kümnendarvuks väljale $A_1^{L_1}$	-
Teisendada kahendarvuks	CVB	4P	RX	Pakitud kümnendarv väljalt A^8 teisendatakse fikskoma arvuks registrisse R	-
Teisendada kümnendarvuks	CVD	4E	RX	Pikskoma arv registrist R teisendatakse pakitud kümnendarvuks väljale A^8	-
Redigeerida	ED	DE	SS'	Pakitud kümnendarv(ud) väljalt A_2^L redigeeritakse (vt. lk. 55) väljale A_1^L	a
Redigeerida ja märkida	EDMK	DP	SS'	Pakitud kümnendarv(ud) väljalt A_2^L redigeeritakse (vt. lk. 55) väljale A_1^L; registrisse 1 märgitakse esimest tüvenumbrit sisaldava baidi aadress	a

Muud direktiivid

Tehte nimetus	Mnem. kood	Arv-kood	For-maat	Tehte sisu	CC
Kodeerida	TR	DC	SS'	L baiti väljalt A_2^L kodeeritakse väljale A_1^L: $(A_2+(A_1+n)\|^1) \rightarrow A_1+n\|^1$ $(n=0,1,\ldots,L-1)$	
Kodeerida ja kontrollida	TRT	DD	SS'	Kontrollitakse järjest baite $(A_2+(A_1+n)\|^1)$ $(n=0,\ldots,L-1)$ kuni leitakse nullist erinev; kui ei leidu, siis $\emptyset \rightarrow$ CC; kui leidub $(n=n_o)$, siis A_1+n_o ja $(A_2+(A_1+n_o)\|^1)$ registritesse 1 ja 2 ning $1\rightarrow$CC (kui $n_o<L-1$) või $2\rightarrow$CC (kui $n_o=L-1$)	⇐
Panna programmi mask	SPM	Ø4	RR'	PSW bittidesse 34-35 (CC) ja 36-39 (PM) saadetakse vastavalt (R^1) bitid 2-3 ja 4-7	⇐
Kontrollida maski järgi	TM	91	SI	Kontrollitakse baidi (A^1) neid bitte, kus maskis I ühed; $\emptyset\rightarrow$CC kui kõik nullid, $3\rightarrow$CC kui kõik ühed, $1\rightarrow$CC kui ühed ja nullid	⇐
Täita käsk	EX	44	RX	$(R^1)\vee(A+1\|^1) \rightarrow A+1\|^1$; täidetakse käsk (A)	-
Pöörduda SUPERVIISORi poole	SVC	ØA	RR"	Vana PSW baitidesse 2-3 (IC) saadetakse ØI; saadud PSW salvestatakse baitidesse 32-39, baitidest 96-1Ø3 võetakse uus PSW	-

8. Translaatori töö juhtimine

Vaatleme lõpuks veel direktiive, mille abil saab juhtida translaatori tööd ning tema poolt väljastatava transleerimisprotokolli vormistamist. Kõigi nende direktiivide kirjeldused on koondatud lehekülgedel 74-75 toodud tabelisse, täiendavaid kommentaare vajavad sealt nähtavasti vaid kolm direktiivi.

Direktiivi ORG kasutatakse üldiselt siis, kui tekib vajadus ühe ja sellesama mälupiirkonna erinevate struktuuride kirjeldamiseks. Olgu näiteks tarvis sisestada ja töödelda informatsiooni perfokaartidelt, kusjuures kaardipakk sisaldagu läbisegi erineva andmetestruktuuriga kaarte. Iga kaardi esimeses veerus asugu struktuuri määrav tunnus, millele vastavalt töödeldakse kord ühe, kord teise struktuuriga kaarte. Näiteks tabelis

KOOD	DS	CL1	1001
A1	DS	CL5	1006
A2	DS	CL24	1030
A3	DS	CL2Ø	1050
A4	DS	CL3Ø	1080
	ORG	A1	1001
B1	DS	2Ø	1021
B2	DS	2Ø	1041
B3	DS	3Ø	1071
	ORG		1080

on esitatud kaht erinevat struktuuri kirjeldavad direktiivid. Tabeli viimases veerus on näidatud kohaloenduri väärtus pärast iga direktiivi transleerimist (kohaloenduri algväärtus olgu 1000).

Direktiiv LTORG on ainus transleerimisdirektiiv, mida tohib märgendada. Kui programmis ei ole ühtegi direktiivi LTORG, siis paigutatakse kõik literaalid esimese (või ainukese) programmsektsiooni lõppu (sinna pannakse ka need literaalid, mis esinevad programmis pärast viimast direktiivi LTORG).

Direktiivi CNOP kasutatakse rajastamiseks siis, kui vahelejäävatesse baitidesse tuleb kirjutada vajalik arv tühidirektiive NOPR. Operand a (mille väärtus võib olla ainult 0, 2, 4 või 6) näitab, mitmenda baidini tuleb rajastada, operand b (lubatavad väärtused 4 ja 8) aga näitab, kas see bait asub sõnas või topeltsõnas. Seega a ja b lubatavate väärtustepaaride korral toimub rajastamine järgmiselt:

0,4 - järgmise sõna algusesse,

2,4 - järgmise sõna teise poolsõna algusesse,

0,8 - järgmise topeltsõna algusesse,

2,8 - järgmise topeltsõna teise poolsõna algusesse,

4,8 - järgmise topeltsõna teise sõna algusesse,

6,8 - järgmise topeltsõna neljanda poolsõna algusesse.

Kui kohaloendur on juba soovitud väärtusega, siis direktiiv CNOP midagi ei muuda. Kui kohaloenduri väärtus on paaritu, siis enne direktiivi CNOP täitmist suurendab translaator teda ühe võrra.

Transleerimisdirektiivid

Tehte nimetus	Direktiivi		Tehte sisu kirjeldus
	kood	aadressosa	
Pealkirjastada	TITLE	'sümbol(id)'	Antud (kuni 100) sümbolit (olemasolu korral ka etikett) trükitakse transleerimisprotokolli iga leheküljе pealkirjaks (etikett ka kirjetesse)
Määrata formaat	ICTL	a a,b a,b,c	Määratakse lähteteksti formaat perfokaartidel: a (1 ≤ a ≤ 40) on alg-, b (41 ≤ b ≤ 80) lõpp- ja c (2 ≤ c ≤ 40) jätkamispositsiooni number
Kontrollida kaartide numeratsiooni	ISEQ	p,q —	Lõpetatakse (aadressosa puudub) või alustatakse lähteteksti kaartide järjekorranumbrite (mis on veergudes p,...,q) kasvamise kontrollimist
Kopeerida raamat	COPY	etikett	Antud nimega lähtetekst (milles ei tohi olla direktiive COPY,END,PRINT,ICTL,ISEQ,MACRO,MEND) lähtemoodulite teegist lülitatakse COPY asemele
Alustada leheküлge	EJECT	—	Transleerimisprotokolli järgneva osa trükkimist alustatakse uuelt leheküljelt
Jätta rida vahele	SPACE	d , —	Kümnendarv d (≤ 256) määrab vahelejäetavate ridade arvu (kui d puudub või selle asemel koma, siis jäetakse vahele üks rida)

Trükkida	PRINT	üks kuni kolm operandi järgmistest alternatiivsetest paaridest: ON/OFF GEN/NOGEN DATA/NODATA	Operandid määravad trükkimise järgmiselt: ON – * mooduli tekst trükitakse, OFF – mooduli teksti ei trükita, GEN – * makrolaiendid trükitakse, NOGEN – makrolaiendeid ei trükita, DATA – konstandid trükitakse täielikult, NODATA – * trükitakse vaid iga konstandi kaheksa vasakpoolset baiti (16 kuueteistkümnendnumbrit); direktiivi puudumisel kehtivad tärniga variandid
Perforeerida	PUNCH	'sümbol(id)'	Perforeeritakse kuni 60 sümbolit aadressosast
Perforeerida järgm. kaart	REPRO	—	Perforeeritakse kuni 80 järgmise kaardi (kirje) sümbolit
Muuta kohaloenduri väärtust	ORG	suhtavaldis —	Antud avaldise väärtus omistatakse kohaloendurile (avaldise etiketid peavad olema eelnevalt määratud); avaldise puudumisel taastatakse maxväärtus
Määrata literaalide piirkond	LTORG	—	Programmi algusest või eelmisest direktiivist LTORG alates esinenud literaalide piirkonna alguseks määratakse järgmise topeltsõna aadress
Rajastada	CNOP	a,b	Absoluutavaldiste a ja b väärtuste järgi rajastatakse kas poolsõnani, sõnani või topeltsõnani
Lõpetada	END	suhtavaldis —	Lõpetab transleerimise; avaldis määrab mooduli sisendpunkti (puudumisel algab täitmine algusest)

Teated transleerimisel avastatud vigade kohta

(teate numbri ette lisab translaator ASSEMBLE koodi IJQ, translaator ASSEMBLF aga koodi IJY)

Teate number ja tekst	Vea iseloom
ØØ1 DUPLICATION FACTOR ERROR	Kordaja on negatiivne või (literaali korral) null
ØØ2 RELOCATABLE DUPLICATION FACTOR	Kordajaks on suhtavaldis
ØØ3 LENGTH ERROR	Pikkus pole lubatud vahemikust või on käskdirektiivis näidatud suhtavaldise abil
ØØ4 RELOCATABLE LENGTH	Pikkuseks suhtavaldis (DC,DS)
ØØ5 S-TYPE CONSTANT IN LITERAL	Literaalis esineb S-tüüpi konstant
ØØ6 INVALID ORIGIN	Direktiiviga ORG määratud väärtus on vale
ØØ7 LOCATION COUNTER ERROR	Kohaloenduri väärtus ületab $2^{24}-1$
ØØ8 INVALID DISPLACEMENT	Suhtaadressi väärtus on vale
ØØ9 MISSING OPERAND	Direktiivis puudub operand
Ø1Ø INCORRECT REGISTER SPECIFICATION	Registri number on kas määratud vale avaldisega, liiga suur või ei vasta nõuetele
Ø11 SCALE MODIFIER ERROR	Mastaabi väärtus ei kuulu lubatud vahemikku
Ø12 RELOCATABLE SCALE MODIFIER	Mastaap on määratud suhtavaldise abil
Ø13 EXPONENT MODIFIER ERROR	Järgu väärtus ei kuulu lubatud vahemikku

Ø14 RELOCATABLE EXPONENT MODIFIER	Järk on määratud suhtavaldise abil
Ø15 INVALID LITERAL USAGE	Literaali vale kasutamine
Ø16 INVALID NAME	Etikett ei vasta nõuetele
Ø17 DATA ITEM TOO LARGE	Konstandi tüüp või pikkus vale
Ø18 INVALID SYMBOL	Operandiväljas vale etikett
Ø19 EXTERNAL NAME ERROR	Vale välisnimi
Ø2Ø INVALID IMMEDIATE FIELD	Vahetu operand on vale
Ø21 SYMBOL NOT PREVIOUSLY DEFINED	Etikett pole eelnevalt määratud
Ø22 ESD TABLE OVERFLOW	Sektsioonide, ühispiirkondade ning välisnimede arv > 255
Ø23 PREVIOUSLY DEFINED NAME	Etikett juba esines või oli kirjeldatud välisnimena
Ø24 UNDEFINED SYMBOL	Määramata etikett
Ø25 RELOCATABILITY ERROR	Suhtavaldise vale kasutamine
Ø26 TOO MANY LEVELS OF PARENTHESES	Avaldises esineb lubatust enam sulgusid üksteise sees
Ø27 TOO MANY TERMS	Termide arv avaldises > 16
Ø28 REGISTER NOT USED	Direktiivis DROP register, mis direktiiviga USING määramata
Ø29 CCW ERROR	Viga direktiivis CCW
Ø3Ø INVALID CNOP	Vale operand direktiivis CNOP
Ø31 UNKNOWN TYPE	Vale tüüp direktiivis DS (DC)
Ø32 OP-CODE NOT ALLOWED TO BE GENERATED	Genereerimisel saadud direktiivi kood pole töödeldav
Ø33 ALIGNMENT ERROR	Rajastamisviga

Ø34 INVALID OP-CODE	Vale tehtekood
Ø35 ADDRESSABILITY ERHOH	Aadress väljaspool baasregistrite tegevuspiirkonda
IJQØ36 NO OPERAND ALLOWED IJYØ36 OPERAND FIELD MUST BE BIANK	Selles direktiivis pole operand lubatud
Ø37 MNOTE STATEMENT	Genereeritud teade
Ø38 ENTRY ERROR	Sisendnimede koguarv > 1ØØ; direktiivis ENTRY kirjeldatud lubamatu etikett
Ø39 INVALID DELIMITER	Eraldaja puudub või on vale
Ø4Ø STATEMENT TOO LONG	Direktiivi pikkus > 127 (255)
Ø41 UNDECLARED VARIABLE SYMBOL	Kirjeldamata muutuv parameeter
Ø42 SINGLE TERM LOGICAL EXPRESSION IS NOT A SETB SYMBOL	Loogilises avaldises ainsana esinev muutuv parameeter pole loogiline parameeter
Ø43 SET SYMBOL PREVIOUSLY DEFINED	Muutuva parameetri korduv määramine
Ø44 VARIABLE SYMBOL SUBSCRIPT EXCEEDS THE DECLARED DIMENSION	Muutuva parameetri dimensioon ei vasta kirjeldatule
Ø45 ILLEGAL SYMBOLIC PARAMETER	Vale püsiparameeter
Ø46 AT LEAST ONE RELOCATABLE Y-TYPE CONSTANT IN ASSEMBLY	Programmis esineb vähemalt üks Y-tüüpi konstant
Ø47 SEQUENCE SYMBOL PREVIOUSLY DEFINED	Suunamisetiketi korduv määramine
Ø48 SYMBOLIC PARAMETER PREVIOUSLY DEFINED OR SYSTEH VARIABLE SYMBOL DECLARED AS SYMBOLIC PARAMETER	Püsiparameetri korduv kirjeldamine; kirjeldatava püsiparameetri nimi ühtib süsteemiparameetri nimega

Ø49 VARIABLE SYMBOL MATCHES A PARAMETER	Muutuva parameetri nimi ühtib püsiparameetri nimega
Ø5Ø INCONSISTENT GLOBAL DECLARATIONS	Sama muutuva parameetri erinev kirjeldamine
Ø51 MACRO DEFINITION PREVIOUSLY DEFINED	Prototüübidirektiivis näidatud makrokäsu kood juba esines
Ø52 NAME FIELD CONTAINS ILLEGAL SET SYMBOL	Parameetri tüüp ei vasta direktiivi SET tüübile
Ø53 GLOBAL DICTIONARY FULL	Globaalsete muutuvate parameetrite koguarv > 4ØØ
Ø54 LOCAL DICTIONARY FULL	Lokaalses sõnastikus pole enam ruumi
Ø55 INVALID NAME IN MACRO INSTRUCTION	Vale etikett makrokäsus
Ø56 ARITHMETIC OVERFLOW	Vale ujukoma konstandi karakteristik või avaldise väärtus
Ø57 SUBSCRIPT EXCEEDS MAXIMUM DIMENSION	Parameetri indeksi väärtus on vale
Ø58 ILLEGAL LTORG	Direktiiv LTORG vale kohas
Ø59 UNDEFINED SEQUENCE SYMBOL	Suunamisetikett pole määratud
Ø6Ø ILLEGAL ATTRIBUTE NOTATION	Karakteristik (kas L, S või I) ei vasta parameetri tüübile
Ø61 ACTR COUNTER EXCEEDED	Suunamiste arv on suurem direktiiviga ACTR näidatust
Ø62 GENERATED STRING GREATER THAN 127 (255) CHARACTERS	Genereeritud sõne pikkus on üle 127 (255) sümboli
Ø63 EXPRESSION 1 OF SUB STRING IS ZERO OR MINUS	Alamsõne esimene avaldis on null või negatiivne
Ø64 EXPRESSION 2 OF SUB STRING IS ZERO OR MINUS	Alamsõne teine avaldis on null või negatiivne

Ø65 INVALID OR ILLEGAL TERM IN ARITHMETIC EXPRESSION	Aritmeetilises avaldises esineb vale või lubamatu term
Ø66 UNDEFINED OR DUPLICATE KEYWORD OPERAND OR EXCESSIVE POSITIONAL OPERANDS	Määramata või korduv nimeline parameeter; positsioonilisi parameetreid on prototüübi direktiivis näidatust rohkem
Ø67 EXPRESSION 1 OF SUBST RING GREATER THAN LENGTH OF CHARACTER EXPRESSION	Alamsõne esimene avaldis on suurem sümbolavaldise pikkusest
Ø68 GENERATION TIME DIC TIONARY AREA OVERFLOWED	Genereerimise ajal täidetavad sõnastikud on ületäitunud
Ø69 EXPRESSION 2 OF SUBSTRING GREATER THAN 8 CHARACTERS	Alamsõne teine avaldis on suurem kaheksast
Ø7Ø FLOATING POINT CHA RACTERISTIC OUT OF RANGE	Ujukoma arvu järk ei kuulu lubatud vahemikku
Ø71 ILLEGAL OCCURRENCE OF LCL, GBL OR ACTR STATEMENT	Direktiivide LCL, GBL või ACTR vale järjekord programmis
Ø72 ILLEGAL RANGE ON ISEQ STATEMENT	Direktiivi ISEQ operandid pole lubatud vahemikust
Ø73 ILLEGAL NAME FIELD	Valesti täidetud nimeväli
Ø74 ILLEGAL STATEMENT IN COPY CODE OR SYSTEM MACRO	Lubamatu direktiiv kopeeritavas tekstis või süsteemi makromäärangus
Ø75 ILLEGAL STATEMENT OUTSIDE OF A MACRO DEFINITION	Direktiivi ei tohi kasutada väljaspool makromäärangut
Ø76 SEQUENCE ERROR	Direktiivide vale järjekord
Ø77 ILLEGAL CONTINUATION CARD	Direktiivi vale jätkamine

Ø78 MACRO MNEMONIC OP-CODE TABLE OVERFLOW	Makrokäskude koodide tabeli ületäitumine
Ø79 ILLEGAL STATEMENT IN MACRO DEFINITION	Direktiivi ei tohi kasutada makromäärangus
Ø8Ø ILLEGAL START CARD	Direktiiv START vale kohas
Ø81 ILLEGAL FORMAT IN GBL OR LCL STATEMENTS	Direktiivides GBL või LCL pole operandiks parameeter
Ø82 ILLEGAL DIMENSION SPECIFICATION IN GBL OR LCL STATEMENTS	Direktiivides GBL või LCL on dimensioon valesti näidatud (ei kuulu vahemikku 1 - 255)
Ø83 SET STATEMENT NAME FI ELD NOT A VARIABLE SYMBOL	Direktiivi SET nimeväli ei sisalda parameetrit
Ø84 ILLEGAL OPERAND FIELD FORMAT IN CONDITIONAL ASSEMBLY STATEMENT	Direktiivi operandiväli on valesti kirjutatud
Ø85 INVALID SYNTAX IN EXPRESSION	Avaldises on süntaksiviga
Ø86 ILLEGAL USAGE OF SYSTEM VARIABLE SYMBOL	Süsteemi parameetri ebaõige kasutamine
Ø87 NO ENDING APOSTROPHE	Puudub lõpetav ülakoma
Ø88 UNDEFINED OPERATION CODE	Määramata tehtekood
Ø89 INVALID ATTRIBUTE NOTATION	Atribuudi vale kasutamine
Ø9Ø INVALID SUBSCRIPT	Vale indeks
Ø91 INVALID SELF DEFINING TERM	Vale konstantterm
Ø92 INVALID FORMAT FOR VARIABLE SYMBOL	Valesti kirjutatud parameeter
Ø93 UNBALANCED PAREN THESES OR EXCESSIVE LEFT PARENTHESES	Viga sulgude kasutamisel

Ø94 INVALID OR ILLEGAL NAME OR OPERATION IN PROTOTYPE STATEMENT	Vale või lubamatu nimi või operatsioonikood prototüübi direktiivis
Ø95 ENTRY TABLE OVERFLOW	Sisendnimede arv > 1ØØ
Ø96 MACROINSTRUCTION OR PROTOTYPE OPERAND EXCEEDS 127 (255) CHARACTERS IN LENGTH	Makrokäsu või prototüübi direktiivi operandi pikkus ületab 127 (255) sümbolit
Ø97 INVALID FORMAT IN MACRO INSTRUCTION OPERAND OR PROTOTYPE PARAMETER	Viga makrokäsu operandis või prototüübi direktiivi parameetris
Ø98 EXCESSIVE NUMBER OF OPERANDS OR PARAMETERS	Operandide või parameetrite arv ületab 1ØØ (2ØØ)
Ø99 POSTIONAL MACRO INSTRUCTION OPERAND, PROTOTYPE PARAMETER OR EXTRA COMMA FOLLOWS KEYWORD	Positsiooniliste ja nimeliste operandide või parameetrite järjekord on rikutud
1ØØ STATEMENT COMPLEXITY EXCEEDED	Liiga keeruline direktiiv
1Ø1 EOD ON SYSIPT OR SYSIN	Puudub direktiiv END
1Ø2 INVALID OR ILLEGAL ICTL	Vigane direktiiv ICTL
1Ø3 ILLEGAL NAME IN OPERAND FIELD OR COPY CARD	Vigane etikett direktiivi COPY operandiväljas
1Ø4 COPY CODE NOT FOUND	Kopeeritav tekst puudub
1Ø5 EOD ON SOURCE STATEMENT LIBRARY	Andmete lõpp teegi makromäärangus
1Ø7 INVALID OPERAND	Vale operand
1Ø8 PREMATURE EOD	Arvuti (translaatori) viga
1Ø9 PRECISION LOST	Täpsuse kadumine
11Ø EXPRESSION VALUE TOO LARGE	Avaldise väärtus ei kuulu vahemikku $-2^{24} \div 2^{24}-1$

Mnemoonilised koodid

Kood	Vastav ingliskeelne fraas	Leheküig
A	Add	62
AD	Add normalized long	64
ADR	Add normalized long Registers	64
AE	Add normalized short	64
AER	Add normalized short Registers	64
AH	Add Halfword	62
AL	Add Logical	62
ALR	Add Logical Registers	62
AP	Add Packed decimal	63
AR	Add Registers	62
AU	Add Unnormalized short	64
AUR	Add Unnormalized short Registers	64
AW	Add unnormalized long	64
AWR	Add unnormalized long Registers	64
B	Branch	69
BAL	Branch And Link	68
BALR	Branch And Link to Register	40, 68
BC	Branch on Condition	68
BCR	Branch on Condition to Register	68
BCT	Branch on CounT	68
BCTR	Branch on CounT to Register	68
BE	Branch on Equal	69
BH	Branch on High	69
BL	Branch on Low	69
BM	Branch on Minus	69
BM	Branch on Mixed	69
BNE	Branch on Not Equal	69
BNH	Branch on Not High	69
BNL	Branch on Not Low	69
BNM	Branch on Not Minus	69
BNP	Branch on Not Plus	69

DS	Define Storage	39
DSECT	identify Dummy SECTion	42 - 43
ED	EDit	55, 70
EDMK	EDit and MarK	55, 70
EJECT	EJECT	74
END	END	75
ENTRY	ENTRY	45
EQU	EQUate	46
EX	EXecute	58, 71
EXTRN	EXTeRNal	45
HDR	Halve long Register	63
HER	Halve short Register	63
IC	Insert Character	61
ICTL	Input format ConTroL	74
ISEQ	Input SEQuence checking	74
L	Load	61
LA	Load Address	61
LCDR	Load Complement long Register	60
LCER	Load Complement short Register	60
LCR	Load Complement Register	60
LD	Load long	61
LDR	Load long Register	59
LE	Load short	61
LER	Load short Register	59
LH	Load Halfword	61
LM	Load Multiple	54, 61
LNDR	Load Negative long Register	59
LNER	Load Negative short Register	59
LNR	Load Negative Register	59
LPDR	Load Positive long Register	59
LPER	Load Positive short Register	59
LPR	Load Positive Register	59
LR	Load Register	59
LTDR	Load and Test long Register	59
LTER	Load and Test short Register	59
LTORG	LiTeral ORiGin	73, 75

LTR	Load and Test Register	59
M	Multiply	62
MD	Multiply long	63
MDR	Multiply long Registers	63
ME	Multiply short	63
MER	Multiply short Registers	63
MH	Multiply Halfword	62
MP	Multiply Packed decimal	63
MR	Multiply Registers	62
MVC	MoVe Characters	60
MVI	MoVe Immediate	60
MVN	MoVe Numerics	60
MVO	MoVe with Offset	60
MVZ	MoVe Zones	60
N	aNd	65
NC	aNd Characters	65
NI	aNd Immediate	65
NOP	No OPeration	69
NOPR	No OPeration to Register	69
NR	aNd Registers	65
O	Or	65
OC	Or Characters	65
OI	Or Immediate	65
OR	Or Registers	65
ORG	ORiGin	72, 75
PACK	PACK	70
PRINT	PRINT optional data	75
PUNCH	PUNCH	75
REPRO	REPROduce	75
S	Subtract	62
SD	Subtract normalized long	64
SDR	Subtract normalized long Registers	64
SE	Subtract normalized short	64
SER	Subtract normalized short Registers	64
SH	Subtract Halfword	62
SL	Subtract Logical	62

SLA	Shift Left single Arithmetic	67
SLDA	Shift Left Double Arithmetic	55, 67
SLDL	Shift Left Double Logical	67
SLL	Shift Left single Logical	67
SLR	Subtract Logical Registers	62
SP	Subtract Packed decimal	63
SPACE	SPACE	74
SPM	Set Program Mask	71
SR	Subtract Registers	62
SRA	Shift Right single Arithmetic	67
SRDA	Shift Right Double Arithmetic	55, 67
SRDL	Shift Right Double Logical	67
SRL	Shift Right single Logical	67
ST	STore	61
START	START	42
STC	STore Character	61
STD	STore long	61
STE	STore short	61
STH	STore Halfword	61
STM	STore Multiple	54, 61
SU	Subtract Unnormalized short	64
SUR	Subtract Unnormalized short Registers	64
SVC	SuperVisor Call	58, 71
SW	Subtract unnormalized long	64
SWR	Subtract unnormalized long Registers	64
TITLE	TITLE	74
TM	Test under Mask	71
TR	TRanslate	57, 71
TRT	TRanslate and Test	57, 71
UNPK	UNPacK	70
USING	USING	40 - 41
X	eXclusive or	65
XC	eXclusive or Characters	65
XI	eXclusive or Immediate	65
XR	eXclusive or Registers	65
ZAP	Zero and Add Packed decimals	60

S i s u k o r d

РУКОВОДСТВО К ПРОГРАММИРОВАНИЮ ДЛЯ ЕС-1022 НА ЯЗЫКЕ АССЕМБЛЕР. Составители Андрес Я э г в р и Юхо К а а з и к. На эстонском языке. Тартуский государственный университет. ЭССР, г. Тарту, ул. Юликооли, 18. Vastutav toimetaja K. Aaremaa. Paljundamisele antud 16.09.1977. Kirjutuspaber 30 x 42 1/4. Trükipoognaid 5,5. Tingtrükipoognaid 5,12. Arvestuspoognaid 3,76. Trükiarv 800. TRÜ trükikoda, ENSV, Tartu, Palsoni t. 14. Tell. nr. 1017. Hind 15 kop.

15 kop.