TARTU ÜLIKOOL
FÜÜSIKA-KEEMIATEADUSKOND
EKSPERIMENTAALFÜÜSIKA JA TEHNOLOOGIA INSTITUUT

ILMAR ANSKO

SPEKTROSKOOPIA PRAKTIKUMI MODERNISEERIMINE

MAGISTRITÖÖ
RAKENDUSFÜÜSIKA ERIALAL

Juhendaja: Ilmar Rammo, füüs.-mat. kand.

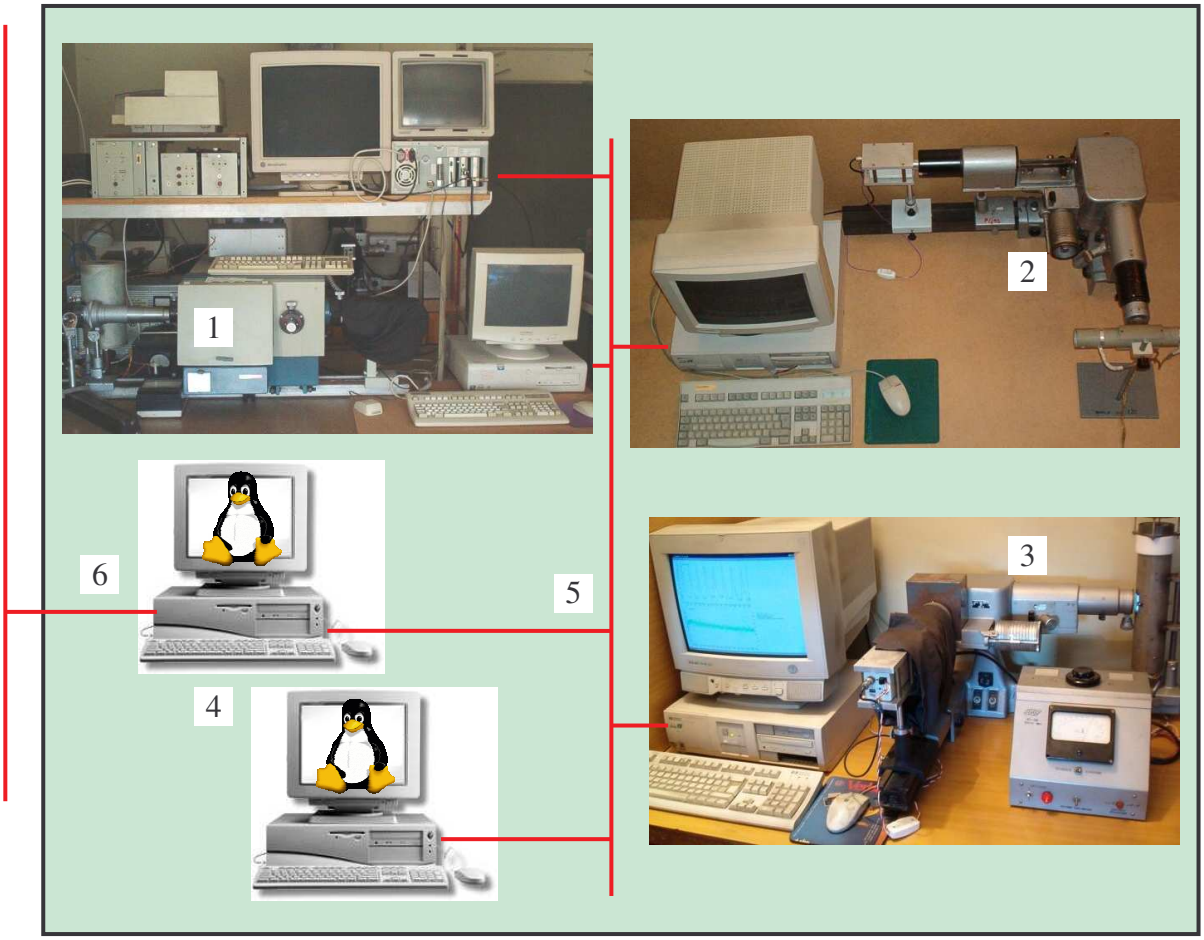
Tartu 2005

Sisukord

1 Sissejuhatus

Spektroskoopia praktikum on olnud kogu pärastsõjaaegse aja Tartu Ülikooli füüsikaosakonna üliõpilaste õppekavas. Tinglikult võib selle praktikumi kujunemise ühe etapi lõppu siduda L. Tuvikese "Spektroskoopia praktikumi tööjuhendid" ilmumisega aastal 1969 [9]. Samaks ajaks oli välja kujunenud ka praktikumis kasutatavate spektraalseadmete park. Pärast seda töötas praktikum stabiilselt enam kui kolmkümmend aastat.

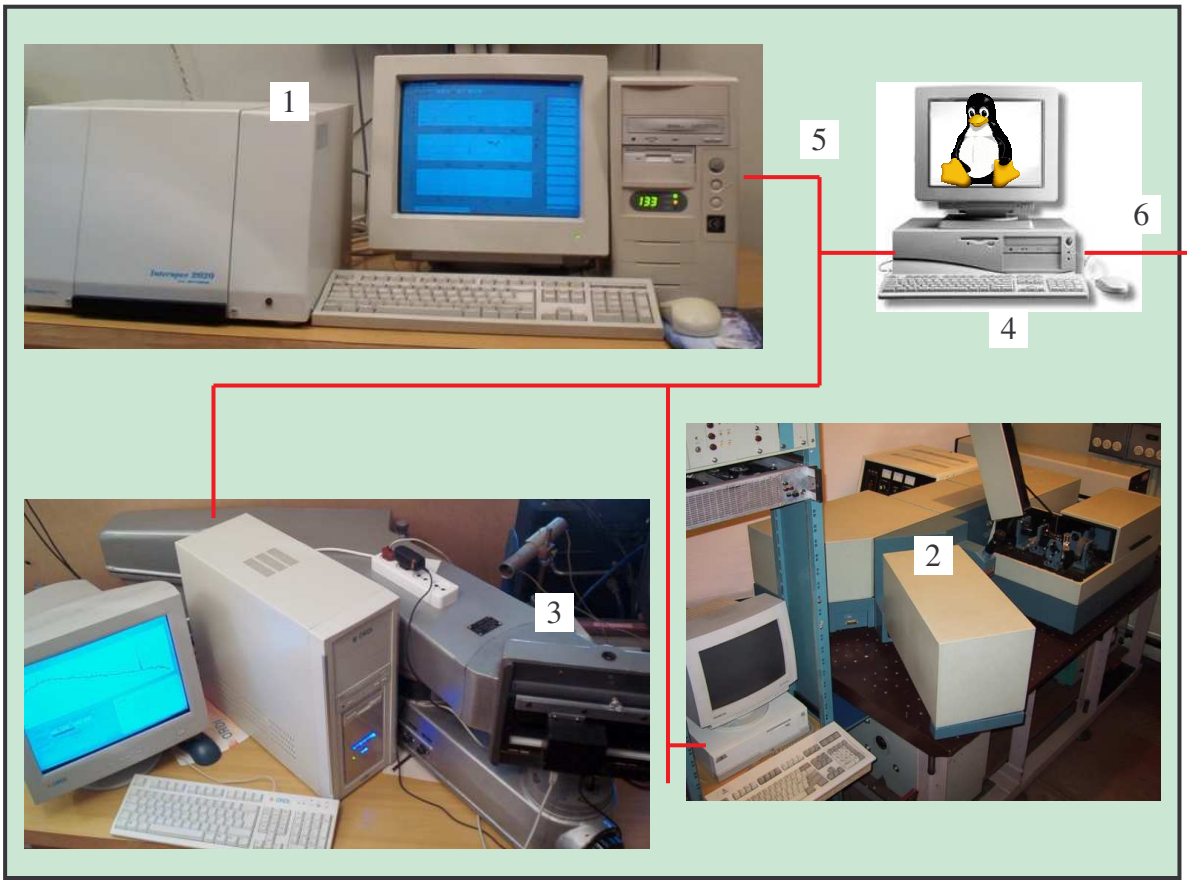
1998. a. püstitas Eksperimentaalfüüsika ja tehnoloogia instituudi juhtkond ülesande spektroskoopia praktikumi moderniseerimisest. Sellest möödunud aja jooksul on praktikum täienenud kahekordse võremonokromaatoriga DFS-52 (valmistatud spetsiaalselt kombinatsioonhajumise spektrite mõõtmiseks), mille kinkis Tartu Ülikoolil füüsika osakonnale a/s "Desintegraator" Tallinnas. Lisaks on praktikumi tarvis ostetud võremonokromaator MDR-



Joonis 1.1. Mõõteaparatuur ruumis 226
1 - monokromaator MDR-23, 2 - Lummeri-Gehrcke plaadiga monokromaator UM-2, 3 - Fabry'-Perot etaloniga monokromaator UM-2, 4 – serverarvutid, 5 – praktikumiruumi kohalik arvutivõrk, 6 – füüsikahoone arvutivõrk

23 ja Tartu firmas “Estla Ltd” valmistatud Fabry-Perot’ etalon. Praktikumis võeti kasutusele (kahasse gaaslahenduslaboriga) Fourier’ spektromeeter (valmistatud Tartu firmas “Interspectrum Ltd”). 2004 a valmis Füüsika osakonna tellimusel Tallinna firmas “AS Laser Diagnostic Instrument“ CCD-kaamera koos programmvarustusega, millega asendati spektrograafi ISP-28 senine fotograafiline registreerimissüsteem.

Käesoleva magistritöö ülesandeks oli välja töötada ja realiseerida (nii programmide kui seadmete tasemel) automaatsed andmekogumissüsteemid töödele “Ramani hajumise uurimine”, “Mõnede lihtsate aatomite spektrite struktuuri uurimine”, “Kiirgusspektrite registreerimine fotoelektrilisel meetodil” ja “Spektrijoone ülipeenstruktuuri uurimine”. Automaatse andmekogumissüsteemi loomise eesmärk on vähendada üliõpilase mehaanilise töö osa praktikumis ja võimaldada tal rohkem aega kulutada tulemuste mõtestamisele. Samas ei ole teadlikult viidud programmidesse katseandmete automaatset töötlemist, jättes selle osa üliõpilase lahendada. Praeguseks ajaks on kõik spektroskoopia praktikumi tööd arvutijuhitavad. Lisaks ilmus 2002. a spektroskoopia praktikumile mõeldud õppematerjal [7], mis käsitleb põhiliste spektraalseadmete ehitust ja toimimise põhimõtteid.



Joonis 1.2. Mõõteaparatuur ruumis 232
1 - Fourier' spektromeeter, 2 - kombinatsioonhajumisspektromeeter DFS-52, 3 - spektrograaf ISP-28, 4 – serverarvuti, 5 - praktikumiruumi kohalik arvutivõrk, 6 - füüsikahoone arvutivõrk

2 Mõõtekompleks monokromaatoriga MDR-23

2.1 Struktuur ja ülesanne

Mõõtekompleksi aluseks on võremonokromaator MDR-23 koos mõnede toite- ja lisaplokkidega. Käesoleva töö käigus on monokromaatori plokke moderniseeritud, juurde on ehitatud kontrolleriplokk mikroprotsessoriga PIC16F84. Kontrolleriplokk on ühendatud juhtarvutiga RS232-siini abil. Mõõtmisprotsessi juhtimiseks on loodud tarkvara nii Windowsi kui ka Linuxi graafilistele kasutajaliidestele. Mõõtekompleksi baasil on erinevate õppekavade üliõpilastele käigus järgmised praktikumitööd:

a) Tutvumine võremonokromaatori ehitusega,

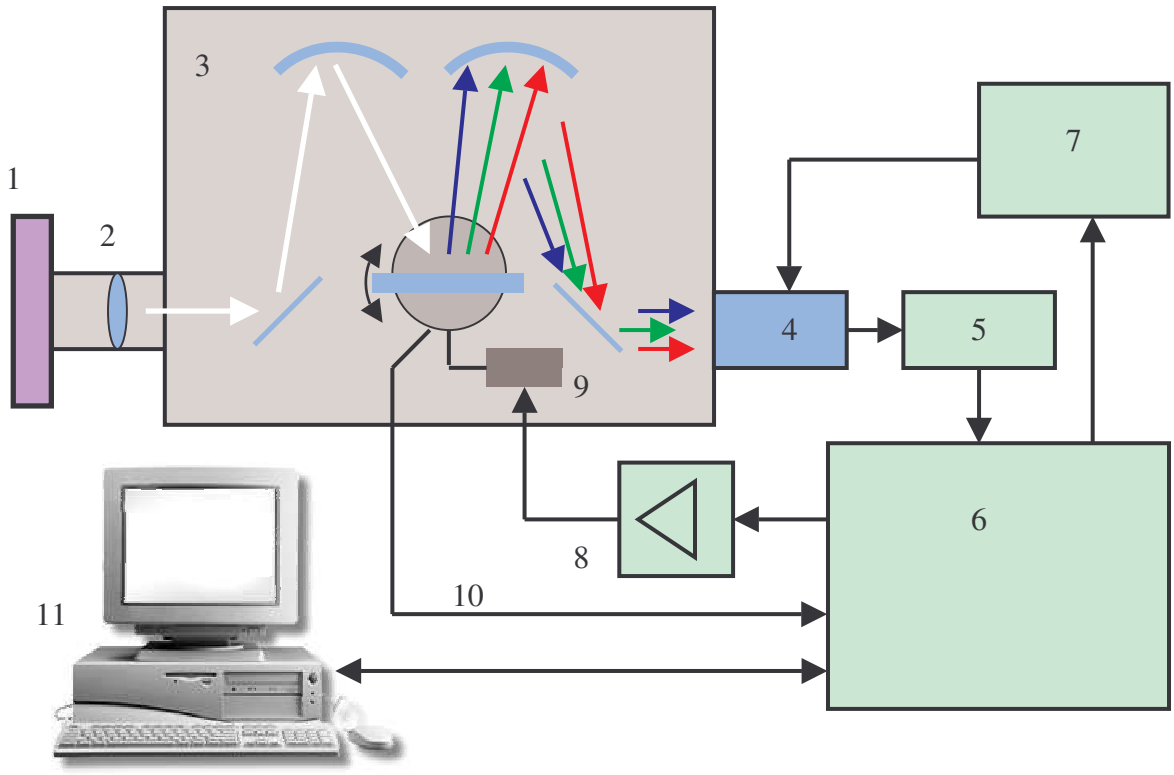
b) Mõnede lihtsate aatomite spektrite struktuuri uurimine,

c) Kiirgusspektri registreerimine fotoelektrilisel meetodil.

Tööd erinevad üksteisest peamiselt uuritava valgusallika poolest, aparaatne ja programmiline osa on kõigi jaoks sama.

2.2 Struktuuriskeem

Mõõtekompleksi struktuuriskeem on toodud joonisel 2.1. Valgus uuritavalt objektilt 1 satub kondensorläätse 2 kaudu monokromaatori 3 sisendpilule. Uuritavateks objektideks on sõltuvalt praktikumitööst madalrõhu-elavhõbelamp, volframniidiga etalonlambi kiirgus magneesiumoksiidiga hajutajalt või kõrgrõhu-elavhõbelambiga ergastatud luminofoor. Monokromaatori väljundpilult langeb valgus fotoelektronkordistile 4. FEK väljundpinge juhitakse võimendiplokki 5, kus asuvad ka FEK ümberlülitatavad koormustakistid. Võimendatud analoogpinge satub edasi kontrolleriplaadile 6, kus ta digitaliseeritakse ja juhitakse mõõtearvutisse. FEK toitepinge saabub plokist 7, mille väljundpinget reguleeritakse kontrolleriplaadilt saabuvate signaalidega. Toitepinge on reguleeritav vahemikus 0..-2200V. Monokromaatori difraktsioonvõret pööratakse samm-mootori 9 abil, mille juhtpinged saabuvad läbi võimendiploki 8 kontrolleriplaadilt. Monokromaatori skaneerimismehhanism on varustatud mõnede kontroll- ja kaitseanduritega, millede signaalid 10 juhitakse samuti kontrolleriplokki. Ülejäänud aparatuuri moodustavad toiteplokid monokromaatori ahelate ja lampide tarvis ning juhtarvuti 11. Mõõtmisprotsessi kontrollib juhtarvuti programm koostöös kontrolleriplaadi

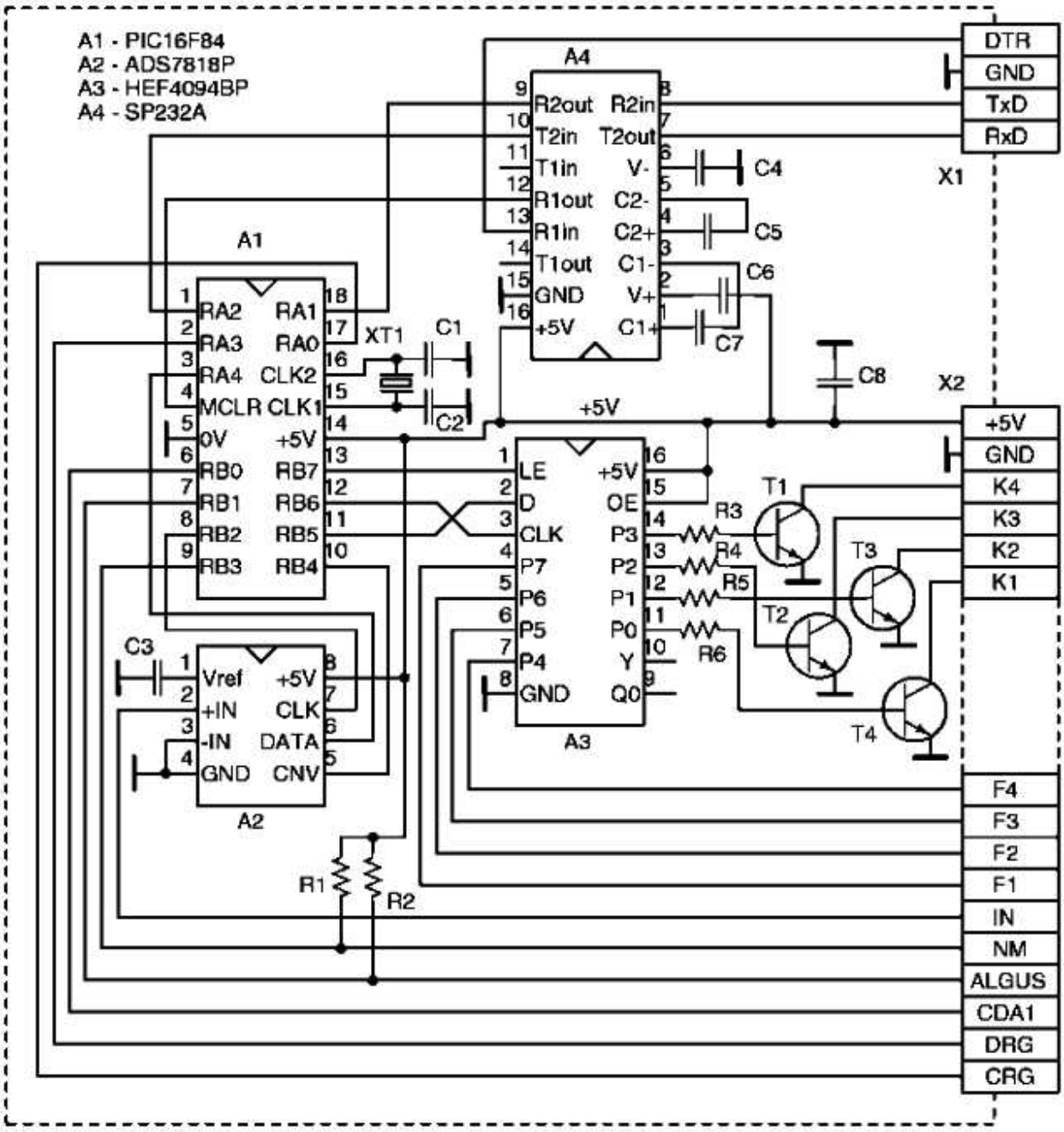


Joonis 2.1. Monokromaatoriga MDR-23 mõõtekompleksi struktuuriskeem.
1 - uuritav valgusallikas, 2 - kondensorlääts, 3 - monokromaator MDR-23, 4 - fotoelektronkordisti (FEK), 5 - FEK eelvõimendi, 6 - kontrolleriplaat, 7 - FEK kõrgepingeplokk, 8 - samm-mootori võimendiplokk, 9 - samm-mootor, 10 - reeperlülitite signaalid, 11 - juhtarvuti

protsessoriga. Side kontrolleriplaadi ja juhtarvuti vahel toimub RS232-protokolli abil. Kontrolleriplaat ja juhtprogrammid on kirjeldatud järgnevates punktides.

2.3 Kontrolleriplaat

Kontrolleriplaadi põhimõtteskeem on toodud joonisel 2.2. Ploki peamisteks komponentideks on mikrokontroller A1 (PIC16F84), analoog-digitaalmuundur A2 (ADS7818), nihkeregister A3 (HEF4094) ja RS232-siini draiver A4 (SP232). PIC-kontrolleril on lisaks toite-, taktgeneraatoriviikudele 13 sõltumatut väratit RA0..RA4, RB0..RB7, mis on programmeeritud vastavalt funktsioonidele sisenditeks ja väljunditeks. Signaalide nimetused ja kirjeldused on toodud tabelis 2.1. Analoog-digitaalmuunduri ADS7818 sisendile 2 saabub (plokist 5 joonisel 2.1) fotoelektronkordisti võimendatud väljundsignaal. Muundamiskäsk ja 12-bitise tulemuse lugemine toimub PIC-kontrolleri poolt jadameetodil signaalide CLK, DATA, ja CNV abil. Register A3 on ette nähtud kontrolleriplaadi digitaalväljundite arvu suurendamiseks. 8-bitise sõna lugemine registrisse toimub viikude D, LE ja CLK abil. Registri väljundid P4..P7 kannavad faasipingeid, mis läbi võimendiploki 8 (joonis 2.1) juhitakse monokromaatori samm-mootori



Joonis 2.2. Kontrolleriplaadi põhimõtteskeem

mähistele. Registri A3 väljundid P0..P3 lülitavad elektromagnetiliste releede kaudu ümber fotoelektronkordisti koormustakisteid. Erinevaid koormustakisteid on kolm: 1MΩ, 3MΩ ja 10MΩ, lisaks neile on võimalik neljanda relee abil FEK väljund lühistada. Releed ja koormustakistid asuvad võimendiplokis 5 (joonis 2.1). Viikude RB1 ja RB3 kaudu saabuvad PIC-kontrollerisse A1 signaalid vastavalt skaneerimismehhanismi reeperlülitilt ja nanomeetriandurilt. RS232-protokolli sisend (signaalid juhtarvutilt kontrollerile) on läbi puhvervõimendi A4 ühendatud PIC-kontrolleri viiguga RA1, väljund aga viiguga RA4. RS232

signaal DTR saabub läbi puhvervõimendi PIC-kontrolleri viigule /MCLR. Sellele viigule antava loogilise 0-impulsiga toimub PIC-kontrolleri initsialiseerimine.

2.4 PIC-kontrolleri juhtprogrammi lühikirjeldus

PIC16F84 on firma MicroChip Inc. KMOP-tehnoloogial põhinev perifeeriakontroller, mille põhilised parameetrid on järgmised: RISC-arhitektuuriga protsessor, 1KB EEPROM-programmimälu, 32 baiti operatiivmälu e. registreid, 32 baiti EEPROM-püsimälu, 13 ükshaaval või baidikaupa (5-bitine PORTA ja 8-bitine PORTB) programmeeritavat digitaalset väratit, välise kvartsresonaatoriga või RC-ahelaga taktgeneraator (töösagedusega kuni 4 MHz). Programmeerimiseks kasutatavad tark- ja riistvaralised vahendid on vabavaralised ning nende kirjeldused on võimalik leida Internetist [5].

PIC-kontrolleri juhtprogrammi struktuur on toodud joonisel 2.3 ja täielik tekst lisas I. Järgnevas kirjelduses viidatakse leheküljenumbritele lisas; programmis kasutatud märgendid ning signaalide ja funktsioonide nimetused on kujutatud kaldkirjas. Kontroller seadub lähteolekusse toitepinge sisselülitamisel või viigule /MCLR saabuva reset-impulsi järel. Peale alglaadimist on kõik kontrolleri väratid programmeeritud digitaalsisenditeks. Alates märgendist *Start* (lk ii) toimub väratite programmeerimine vastavalt signaalidele sisenditeks või väljunditeks. Signaali *TxD* väärtuseks seatakse loogiline 1, mis vastab RS232-siini vaikeolekule. Alamfunktsiooniga *ToShiftReg* kirjutatakse nihkeregistrisse A3 (joon. 2.2) väärtus 0x00, millega kõrvaldatakse samm-mootorilt faasipinged. Alates märgendist *Command* algab programmi põhiline töötsükkel. Alamfunktsiooni *RxByte* käigus ootab kontroller RS232-sisendilt juhtbaidi saabumist ja vastavalt selle baidi väärtusele siirdub järgmisele märgendile. Iga käsu täitmisega seotud tegevuse lõpus siirdub programm märgendile *Valmis*, kus saadetakse RS232-siinile *PORTB* väärtus. *PORTB*

viigu nr	nimetus	signaal	kirjeldus
1	RA2	RxD	RS232 siini väljund
2	RA3	DRG	DA-muunduri juhtimine
3	RA4	DATA	AD-muunduri juhtimine
6	RB0	CDA1	DA-muunduri juhtimine
7	RB1	ALGUS	signaal reeperlülitilt
8	RB2	CLK	AD-muunduri juhtimine
9	RB3	NM	signaal 1nm andurlülitilt
10	RB4	CNV	AD-muunduri juhtimine
11	RB5	D	nihkeregistri juhtimine
12	RB6	CLK	"
13	RB7	LE	"
17	RA0	CRG	DA-muunduri juhtimine
18	RA1	TxD	RS232 siini sisend

Tabel 2.1. Signaalid PIC-kontrolleri viikudel

bitid *NM* ja *L6PP* kannavad signaale monokromaatori skaneerimismehhanismi kontroll-lülititelt. Seejärel siirdub programm töötsükli algusesse märgendile *Command*. Kui saabunud bait ei vasta ühelegi juhtkäsule, saadab kontroller RS232-siinile baidi 0x00 ja siirdub ootama uut käsku. Juhtkäsud ja neile vastavad tegevused on järgmised.

'n': programm siirdub märgendile *Nihkereg*, kus loeb RS232-siinilt argumendibaidi. Argument kirjutatakse alamfunktsiooniga *ToShiftreg* registrisse A3 (joon. 2.2).

'v': programm siirdub märgendile *Vasakule*, kus loetakse RS232-siinilt kaks argumendibaiti. Argumendid moodustavad 16-bitise sammude arvu, mille võrra nihutatakse monokromaatori skaneerimismehhanismi. Iga sammu eel inkrementeeritakse muutuja *mootor* väärtust. Selle muutuja kaks noorimat bitti omandavad seega tsükliliselt väärtusi 00, 01, 10 ja 11. Vastavalt sellele väärtusele tagastab alamfunktsioon *Faas* ühe neljast võimalikust faasipingete kombinatsioonist, mis seejärel nihkeregistri A3 ja võimendite 8 (joonis 2.2) kaudu juhitakse mootori mähistele. Monokromaatori skaneerimismehhanism liigub lainepikkuse kasvamise suunas. Alamfunktsiooniga *Dectime* kahandatakse ühele sammule kuluvat aega sujuvalt kuni miinimumväärtuseni, mis garanteerib skaneerimise suurima võimaliku kiirusega (umbes 3 nm sekundis). Iga sammu järel kontrollitakse sisendsignaali *L6PP*, mis omandab loogilise 0-väärtuse, kui skaneerimismehhanism on sattunud väljapoole lubatud piire. Sellisel juhul sammumine katkestatakse ja programm siirdub kohe märgendile *Valmis*. Juhtarvutile tagastatav *PORTB* väärtus sisaldab sealjuures kehtivaid signaale *NM*, *L6PP*.

'p': programm siirdub märgendile *Paremale*. Nagu eelmine, ainult muutuja *mootor* väärtust inkrementeerimise asemel dekrementeeritakse. Selle tulemusel muutub samm-mootori faasipingete järjekord eelmisega võrreldes vastupidiseks ja monokromaatori skaneerimismehhanism liigub lainepikkuse kahanemise suunas.

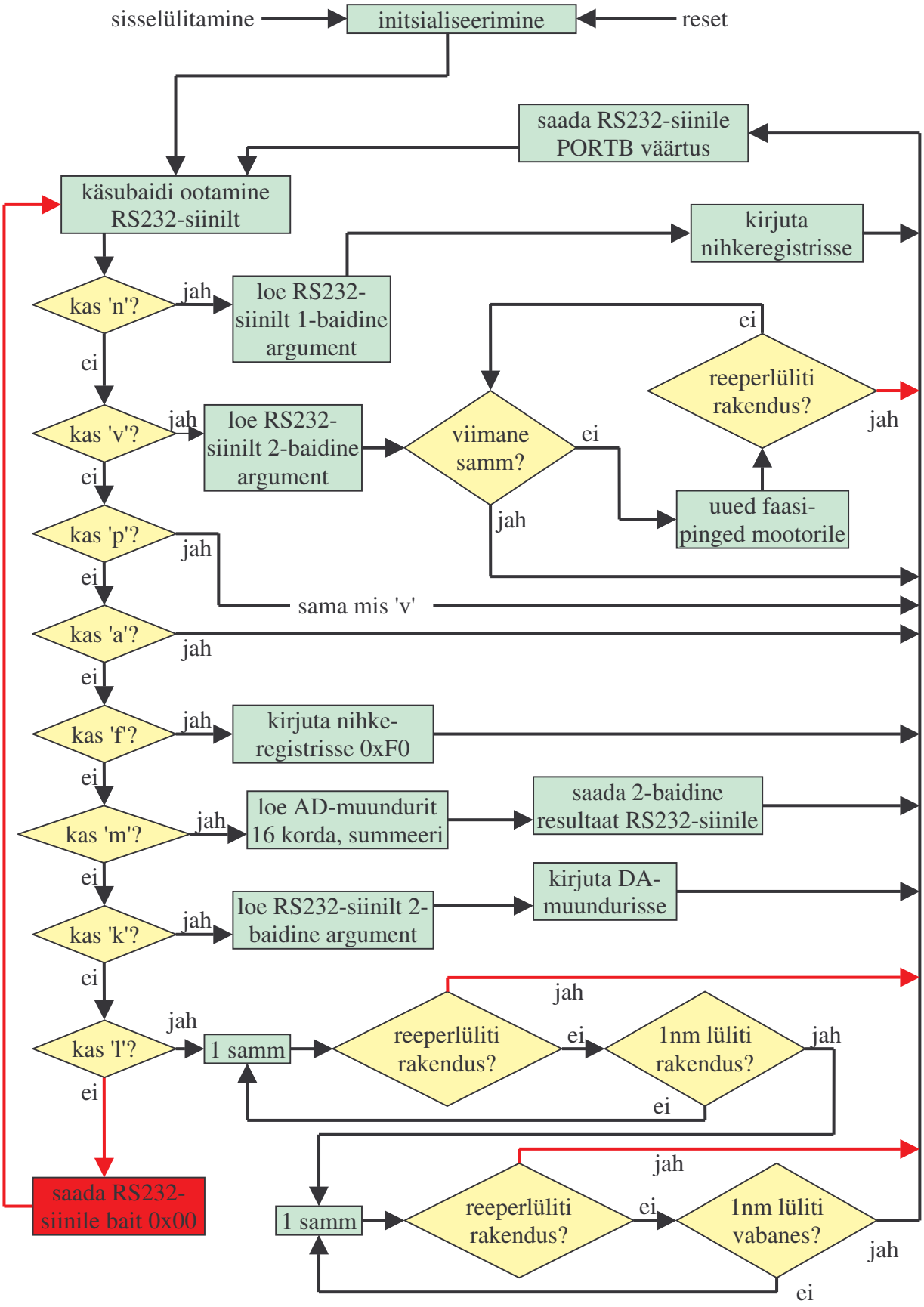
'a': programm siirdub märgendile *Andurid* ja sealt kohe märgendile *Valmis*, tagastades RS232-siinile *PORTB* väärtuse.

'f': programm siirdub märgendile *FaasMaha*. Nihkeregistrisse A3 kirjutatakse väärtus, mis vastab samm-mootori faasipingete puudumisele.

'm': siirdumine märgendile *M66tmine*. Toimub 12-bitise väärtuse lugemine AD-muundurist A2 (joon. 2.2) 16 korda järjest alamfunktsiooni *ADRead* abil, kusjuures tulemus summeeritakse kahebaidiliselt muutujatesse *low1, high1*. Seejärel saadetakse muutujate *low1* ja *high1* väärtused RS232-siinile ning siirdutakse märgendile *Valmis*.

'k': programm siirdub märgendile *K6rgePinge*. RS232-siinilt loetakse 2 argumendibaiti, mis moodustavad 12-bitise sisendväärtuse kõrgepingeploki 7 (joon. 2.2) DA-muunduri jaoks. See

väärtus kirjutatakse DA-muundurisse jadameetodil signaalide *KPDATA*, *KPCLOCK* ja *KPLOAD*



Joonis 2.3. PIC-kontrolleri programmi plokkskeem

abil.

'l': programm siirdub märgendile *Reeper*. Seal toimub monokromaatori skaneerimismehhanismi nihutamine sammhaaval lainepikkuse kahanemise suunas, kuni sisendile *NM* saabub kontroll-lülitilt loogiline 0-nivoo. Seejärel nihutatakse skaneerimismehhanismi lainepikkuse kasvu suunas, kuni sisendile *NM* tekib 1-nivoo. Selle protseduuri tagajärjel on skaneerimismehhanism seatud kindlasse asendisse monokromaatori esipaneelil asuva numbrilise skaalaindikaatori suhtes. Iga sammu järel kontrollitakse ka reeperandurit *L6PP*, mille rakendumisel protseduur katkestatakse.

2.5 Juhtarvuti programmid

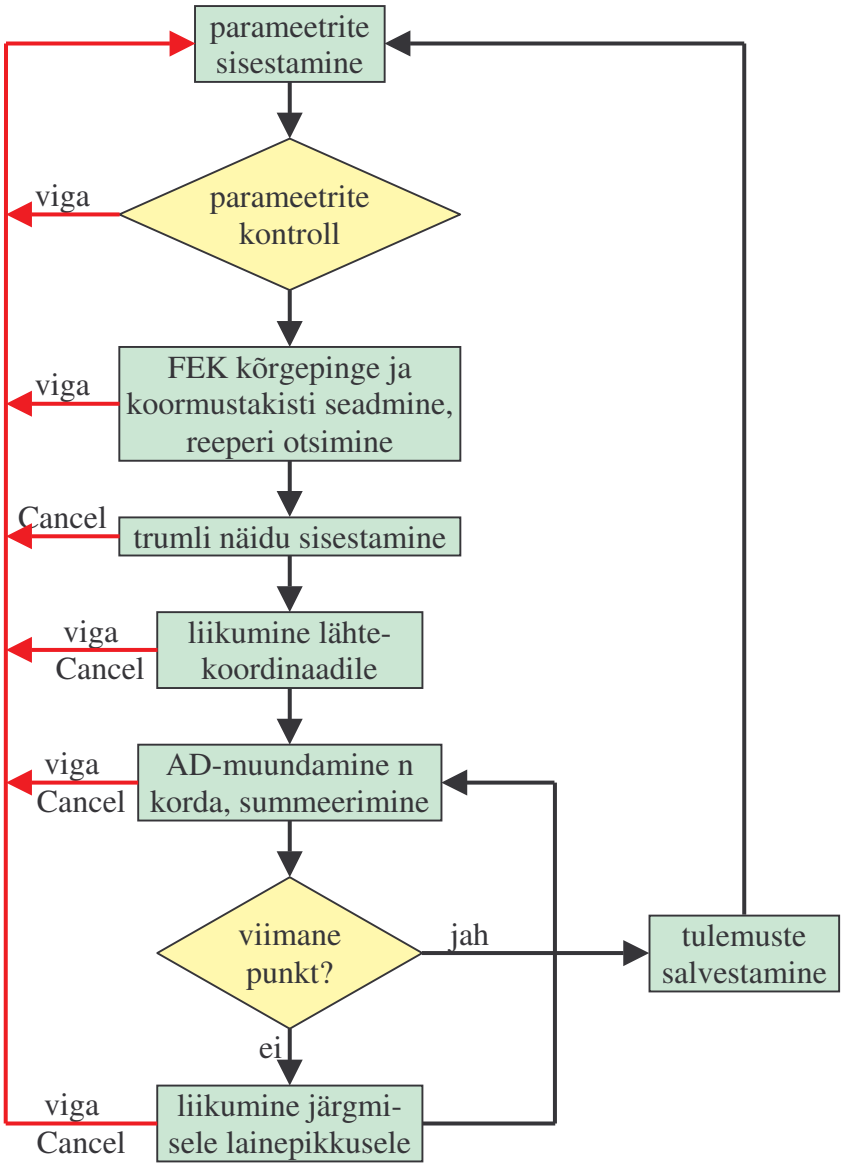
Juhtprogramm on realiseeritud nii Windowsi kui ka Linuxi graafilise keskkonna jaoks. Skaneerimisprotsess toimub joonisel 2.4 näidatud plokkskeemi järgi. Juhtprogrammide lähtetekstid on toodud lisas I. Windows-programm on kompileeritav nii Borlandi Delphi-keskkonnas kui ka vabavaralise kompilaatoriga Free Pascal [3]. Linux-programm on kompileeritud Free Pascali abil.

2.5.1 Windows-programmi lühikirjeldus

Programm vastab põhimõttelisele struktuuriskeemile, nagu toodud lisas IV, kus on ka selgitatud Windows-programmeerimise põhimõisted nagu aken (Window), aknafunktsioon (Window Procedure), sündmused (Events), teated (Messages). Järgnevas kirjelduses on funktsioonide ja muutujate nimetused kujutatud kaldkirjas. Programmi peamise akna aknafunktsioon on *MainProc()* (lk xxviii). Peaakna identifikaator on *g_hwMain* ja aknafunktsiooni piires ka *DlgWin*. Kompilaatoridirektiiviga *{$R m95.res}* (lk x) lisatakse ressursifail m95.res, milles paiknevad dialoogiakende ja peaakna menüü kirjeldused. Ressursifail on kompileeritud lähtetekstist m95.rc (vt lisa I) firma Borland vabavaralise ressursikompilaatori brcc32 abil.

Esimene teade, mille operatsioonisüsteem peaakna aknafunktsioonile saadab, on *WM_CREATE* (lk xxix). Selle teate töötlemise käigus püütakse lugeda failist mdr.ini (vt. lisa I) skaneerimise vaikimisi-parameetrid ja initsialiseeritakse mitmesugused programmisisesed muutujad. Peaakna sulgemise eel saadetakse tema funktsioonile teade *WM_DESTROY*, mille peale toimub operatsioonisüsteemi ressursside vabastamine ja väljumine teatetsüklist funktsiooniga *PostQuitMessage()*.

Kui kasutaja teeb valiku peaakna menüüs, saadetakse aknafunktsioonile teade *WM_COMMAND*, mille parameeter *wParam* määrab menüüelemendi. Osad menüüelemendid on programmi



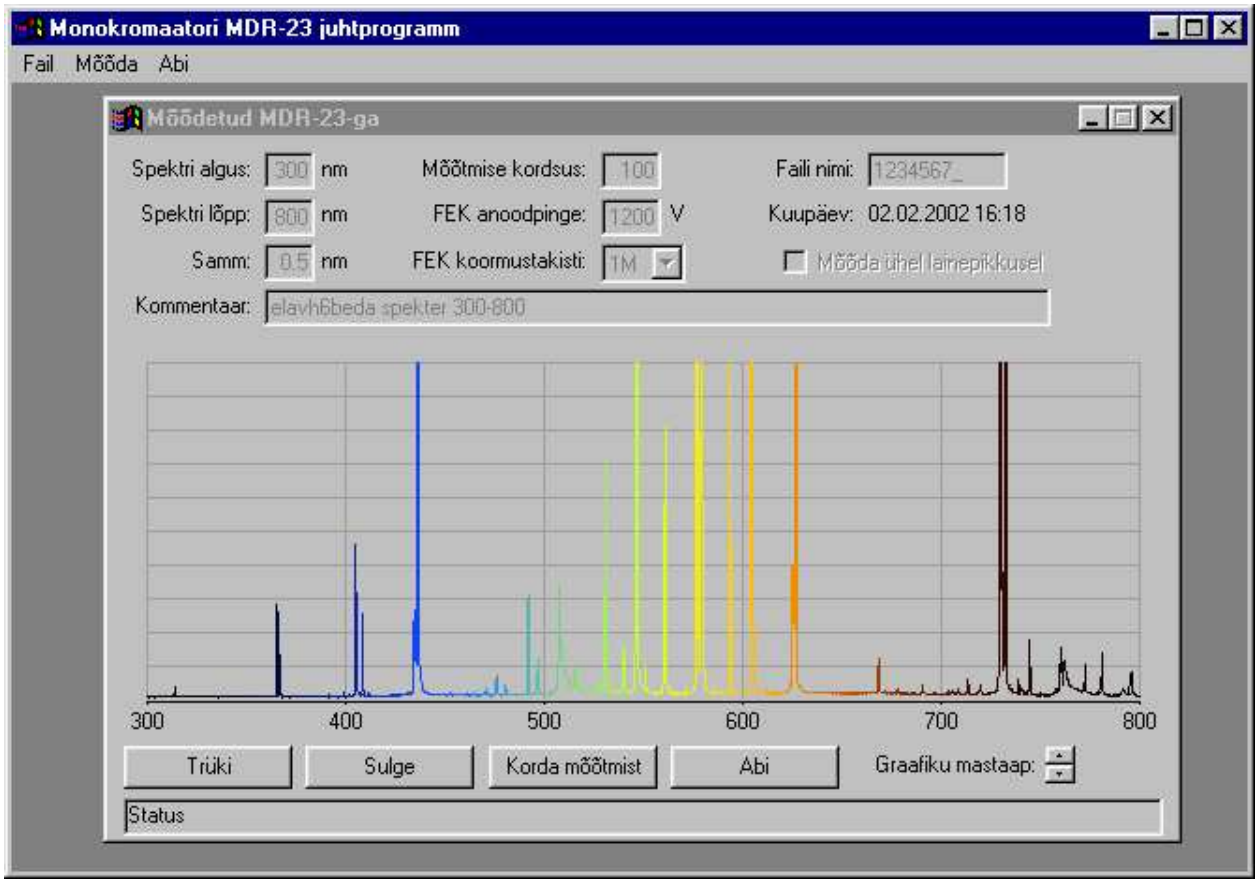
Joonis 2.4. Mõõtmistsükli plokkskeem

erinevates faasides mitteaktiivsed ja neid kasutaja valida ei saa. Menüüvalikuid keelatakse ja lubatakse funktsiooniga *EnableMenuItem()*. Kui valida *Fail*-menüüst element *Ava*, saadetakse aknafunktsioonile teade *WM_COMMAND* parameetriga *wParam*=101 (lk xxviii). Funktsiooni *OpenFileName()* koosseisus olev API-funktsioon *GetOpenFileName* (lk xv) toob ekraanile Windowsi tavapärase failivalikuakna. Valiku tulemus (faili nimi) antakse edasi funktsioonile *LoeSisse()*, mis püüab lugeda sellest failist katseparameetreid. Mdr_95 katseandmeid hoitakse ASCII-tekstifailis, mille struktuur on toodud lisas I. Lugemisvea korral, või kui faili formaat ei vasta oodatule, tagastab *LoeSisse()* väärtuse *FALSE*, aknafunktsioon kujutab *MessageBox()* abil veateate ja tagastab juhtimise operatsioonisüsteemile. Kui katseandmete lugemine failist õnnestub, luuakse uus mittemodaalne dialoogiaken käsuga *CreateDialog()*. Dialoogiakna elementide paigutus asub ressursifailis m95.res ja tema aknafunktsioon on *AvaProc()* (lk xxvi).

Esimene teade, mis uue dialoogi aknafunktsioonile saadetakse, on *WM_INITDIALOG* (lk xxviii). Selle teate töötlemise käigus arvutatakse kõigepealt graafiku joonistamise konstandid funktsiooniga *ArvutaParameetrid()* ja kirjutatakse mõõtmise parameetrid dialoogiakna lahtritesse funktsiooniga *T2idaLahtrid()* (lk xviii). Järgmine oluline teade, mille operatsioonisüsteem aknafunktsioonile saadab, on *WM_PAINT*. Selle teate töötlemisel on programmil lubatud dialoogiakna kliendipiirkonda GDI-funktsioonide (GDI - Graphical Device Interface) abil joonistada (lk xxvi). Joonistamise alguses tuleb kutsuda välja API-funktsioon (API - Application Program Interface) *BeginPaint()* ja lõpus *EndPaint()*. Nende funktsioonide vahel kujundatakse akna kliendipiirkonda koordinaatvõrgustik ja mõõtmistulemuse graafik API-funktsioonide *MoveTo()*, *LineTo()*, *SetPixel()* ja *TextOut()* abil. Tegelik joonistamine dialoogiaknale toimub pärast seda, kui aknafunktsioon on operatsioonisüsteemile juhtimise tagastanud.

Graafiku mastaapi muudetakse Windowsi standardse elemendi *UpDownControl* abil (vt. joon. 2.5), mis loodi teate *WM_INITDIALOG* töötlemise käigus API-funktsiooni *CreateUpDownControl()* abil. See element saadab aknafunktsioonile teateid *WM_VSCROLL*, kus parameetriks *lParam* on elmendi identifikaator *g_hwUpDownAva* (lk xxviii). Selle teate töötlemisel, vastavalt mastaabi muutmise suunale, kasvatatakse või kahandatakse mastaabikordaja *g_AvaHeader.k1* väärtust. Seejärel kutsutakse välja API-funktsioon *InvalidateRect()*, mis teatab operatsioonisüsteemile, et dialoogiakna sisu vajab ülejoonistamist, mis omakorda toob kaasa teate *WM_PAINT* saatmise aknafunktsioonile. *WM_PAINT* töötlemise käigus joonistatakse graafik uuesti, endisest erineva mastaabikordaja väärtusega. Dialoogiakna sulgemisel saadetakse tema aknafunktsioonile viimaks teade *WM_CLOSE*, mille peale programm kutsub välja funktsiooni *DestroyWindow()*.

Kui valida peaakna menüüst *Mõõda→Mõõda*, saadetakse tema aknafunktsioonile *MainProc()* teade *WM_COMMAND* parameetriga *wParam*=201 (lk xxix). Seal täidetakse mõõtmisparameetrid vaikimisi-väärtustega ja saadetakse aknafunktsioonile omakorda teade *WM_COMMAND* parameetriga *wParam*=503 (lk xxix). Selle teate töötlemisel luuakse käsuga *CreateDialog()* uus dialoogiaken sarnaselt eelpoolkirjeldatule, ka dialoogiressurss (st. nuppude, tekstilahtrite jms paigutus) on sama, ainult aknafunktsiooniks on *M66daProc()* (lk xx). Teadete *WM_INITDIALOG*, *WM_CLOSE*, *WM_PAINT* ja *VM_VSCROLL* töötlemine toimub sarnaselt eelmisele. Kui kasutaja on lahtrid tulevase mõõtmise parameetritega täitnud ja vajutab nuppu *Mõõda*, saadetakse aknafunktsioonile teade *WM_COMMAND* parameetriga *wParam*=103. Selle teate töötlemisel (lk xxi) kontrollitakse kõigepealt funktsiooni *GetDlgItemText()* abil nupul olevat teksti. Nimelt mõõtmisprotsessi alates asendatakse nupul tekst "*Mõõda*" tekstiga



Joonis 2.5. Juhtprogrammi ekraanipilt Windows-keskkonnas

"*Katkesta*" (funktsiooniga *SetDlgItemText()*, lk xxiii) ja sellega koos muutub ka nupu funktsioon. Mõõtmiskäsu andmisel hakkab programm kontrollima sisestatud parameetreid ja väljastab ebasobiva väärtuse korral veateate funktsiooniga *MessageBox()*. Kui parameetrite kontroll on lõppenud, avatakse käsuga *AvaPort()* RS232-port ja kui see õnnestub, postitatakse aknafunktsioonile teade *WM_COMMAND* parameetriga *wParam*=505.

Mõõtmisprotsess on organiseeritud järgmiselt. Et tagada põhiprogrammi asünkroonsus ja mitteblokeeruvus, luuakse programmi peamise lõime (*thread*) kõrvale käsuga *CreateThread()* teine, mis suhtleb RS232-pordiga sünkroonsel (blokeeruval) viisil ja saadab programmiaknale teateid funktsiooni *PostMessage()* abil. See võte lubab teise lõime sees kasutada "lineaarset" programmeerimismudelit ja ühtlasi õigustab mõõtmistsükli kujutamise viisi joonisel 2.4. Sellest tuleb aru saada nii, et protsessid toimuvad joonisel 2.4 peamise mõõtmistsükli sees nooltega näidatud suunas ja järjestuses just teise lõime mõttes, samal ajal kui ülejäänud programmi aknafunktsioone kutsutakse välja asünkroonselt ja vahepeal antakse juhtimine tagasi operatsioonisüsteemile.

RS232-port avatakse API-funktsiooniga *CreateFile()* (lk xi). Porti kirjutamine ja pordist lugemine toimub käskudega *WriteFile()* ja *ReadFile()*. Lugemine ja kirjutamine on siin

sünkroonsed (blokeeruvad) operatsioonid. Funktsiooniga *SetCommTimeOut()* (lk x) seatakse maksimaalne aeg, mille jooksul *ReadFile()* pordist vastust ootab. Kui selle aja möödumisel nõutav arv baite pole saabunud, katkestab mõõtmislõim oma tegevuse käsuga *ExitThread()* ja saadab aknafunktsioonile teate *WM_COMMAND* parameetriga *wParam*=509 (lk xxv). Selle peale kuvab aknafunktsioon käsu *MessageBox()* abil veateate ja vastavalt kasutaja valikule (*MessageBox()* tagastab nupule *Retry* vajutamisel konstandi *IDRETRY* ja *Cancel* puhul konstandi *IDCANCEL*) postitatakse aknafunktsioonile teade *WM_COMMAND* parameetriga *wParam*=505 (mõõtmise alustamine) või *wParam*=509 (katkestamine). Mõõtmislõimesid on kokku neli ja igaühele vastab oma lõimefunktsioon (thread-function): *T66Proc0()*, *T66Proc1()*, *T66Proc3()* ja *T66Proc4()* (lk xii..xv). Iga sellise funktsiooni sees toimub tegevus sarnasel põhimõttel: a) kirjutada RS232-porti instruktsioon; b) lugeda vastus; c) vastavalt tulemusele katkestada/jätkata tegevust, saates mõõtmisdialoogi aknafunktsioonile teateid funktsiooni *PostMessage()* abil.

Mõõtmisprotsessi alates luuakse kõigepealt lõim funktsiooniga *T66Proc0()*, mis saadab RS232-pordi kaudu kontrollerile käsud ‘n’ ja ‘k’ koos parameetritega, seades nii FEK koormustakisti ja katoodpinge väärtuse (vt jaotist 2.4). Seejärel lõim lõpetab tegevuse ja saadab mõõtmisdialoogi aknafunktsioonile teate *WM_COMMAND* parameetriga *wParam*=501 (lk xxiv). Aknafunktsioon loob järgmiseks lõime funktsiooniga *T66Proc4()* (lk xiv). Seal saadetakse kontrollerile käsk ‘l’, mille peale monokromaatori skaneerimismehhanism liigub kindlasse, reeperlülititega määratud asendisse. Seejärel kujutatakse ekraanile dialoogiaken, milles on lahter numbri sisestamiseks monokromaatori esipaneelilt ning nupud *OK* ja *Cancel*. Selle dialoogiakna ressurss (elementide paigutus) asub failis m95.res, aknafunktsiooniks on *KoordinaatProc()* (lk xi). Kuna see dialoogiaken on modaalne (st loodud funktsiooniga *DialogBox()*), siis tema ekraaniloleku ajaks lõimefunktsiooni *T66Proc4()* täitmine peatub. Kui kasutaja on numbri sisestanud ja vajutanud dialoogiaknal nuppu *OK*, toimub dialoogiakna hävitamine käsuga *EndDialog()* ja funktsioonile *T66Proc4()* tagastatakse sisestatud lainepikkuse väärtus. *T66Proc4()* arvutab nüüd samm-mootori liikumise suuna ja vajaliku sammude arvu ja saadab kontrollerile käsu ‘p’ või ‘v’ koos parameetritega (vt punkt 2.4). Kontroller saadab käsu täitmise kinnituseks baidi, mis sisaldab skaneerimismehhanismi reeperlülitite asendeid. Lõimefunktsioon kontrollib alati funktsiooniga *ReadFile()* saadud baidi väärtust (so PIC-kontrolleri PORTB väärtust, vt jaotis 2.4) ja kui tuvastab veaolukorra, katkestab tegevuse, saates mõõtmisdialoogi aknafunktsioonile teate *WM_COMMAND* parameetritega *wParam*=509, *lParam*=2. Vea puudumisel saadab *T66Proc4()* mõõtmisdialoogile teate *WM_COMMAND* parameetriga *wParam*=506 (lk xv) ja lõpetab tegevuse. Järgmiseks loob mõõtmisdialoog lõime funktsiooniga *T66Proc1*() (lk xii), mis saadab kontrollerile vaheldumisi käske ‘m’ (AD-muunduri lugemine) ja ‘v’ (skaneerimine). Iga punkti

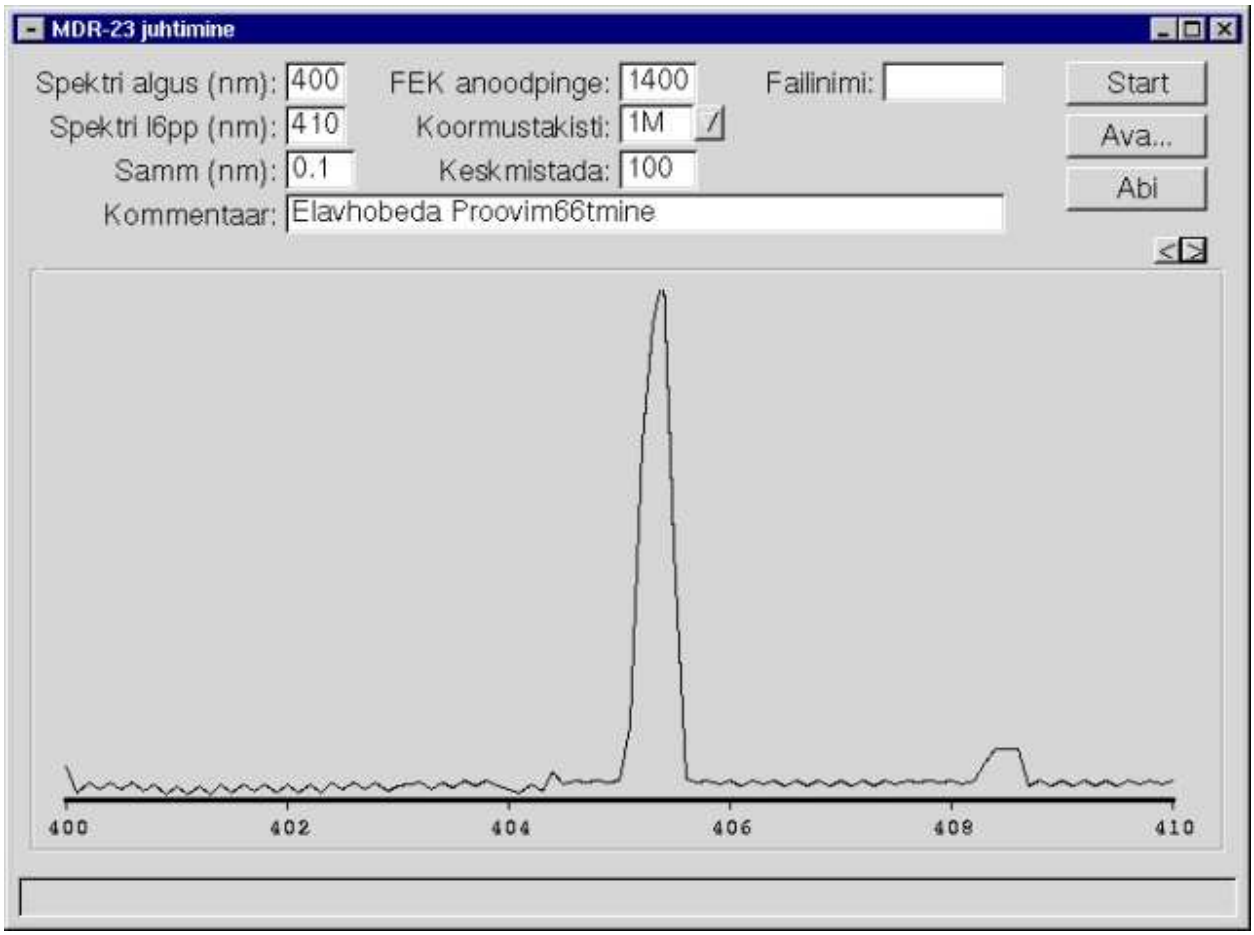
mõõtmise järel saadetakse mõõtmisaknale teade *WM_COMMAND* parameetriga *wParam*=502 (lk xxiv). Mõõtmisdialoog postitab seejärel iseendale teate *WM_PAINT*, mis põhjustab graafikupiirkonna ümberjoonistamise. Pärast viimast katsepunkti saadab lõim mõõtmisaknale teate *WM_COMMAND* parameetriga *wParam*=503 (lk xxiv). Mõõtmistulemused salvestatakse kettafaili funktsiooniga *Salvesta()* (lk xvi) ja luuakse lõim funktsiooniga *T66Proc3()* (lk xiii). Viimane saadab kontrollerile käsu 'k' parameetritega 0,0, kõrvaldades FEK toitepinge, ja lõpetab oma tegevuse, saates mõõtmisaknale teate *WM_COMMAND*, *wParam*=504 (lk xxiv). Sellega on mõõtmisprotsess lõppenud. Mõõtmisdialoogi aken jääb avatuks, identifikaatorile *103* vastava nupu kirjaks seatakse "*Mõõda*" ja ka ülejäänud dialoogiakna elemendid ja programmi peaakna menüüelemendid seatakse funktsioonide *EnableWindow()* ja *EnableMenuItem()* abil olekusse, mis lubavad alustada parameetrite sisestamist uue mõõtmise tarvis.

2.5.2 Linux-programmi lühikirjeldus

Programmi lähtetekst asub lisas I, kompileerimisel on kasutatud Free Pascalit. Programm baseerub GTK (GIMP Toolkit, GIMP - GNU Image Manipulation Program, GNU - rekursiivne akronüüm "GNU's not UNIX"), GDK (General Drawing Kit) ja Xlib ressurssidel. Operatsioonisüsteemi Linux graafiline interfeis kannab nimetust "X-System" või ka lihtsalt "X". X-süsteemiga käib kaasas suur hulk abiteeke (library), mida on võimalik kasutada uute X-rakenduste (st graafilise kasutajaliidesega programmide) loomisel. Programmeerimise hõlbustamiseks ja unifitseerimiseks on loodud mitmeid abiteeke ja -keskkondi, mille hulka kuuluvad ka GTK ja GDK. GTK on defineeritud järgmiselt: multiplatvormiline vahend graafilise kasutajaliidese loomiseks (multi-platform toolkit for creating graphical user interfaces) [4]. GTK kasutab X-süsteemiga suhtlemiseks omakorda GDK funktsioone ja mehhanisme. GDK-s realiseeritakse muuhulgas programmi teatetsükkel (event loop) ja lõimed (multithreading). GTK ja GDK funktsioonid on mõnes mõttes MS Windowsi API-funktsioonide analoogid Linuxi keskkonnas.

X-süsteemi, GDK ja GTK funktsioonide kasutamiseks Free Pascal programmis tuleb vastavad moodulid registreerida *uses*-klausli abil (lk xxxiii).

Programmi alguses (lk xliv) luuakse peaaken käsuga *gtk_window_new()*. Järgnevalt luuakse peaaknasse redigeerimisaknad, nupud jm juhtimiselemendid käskudega *gtk_..._new()*. GTK-programmis ühine aknafunktsioon kõiki juhtelemente ja peaakent puudutavate sündmuste (event) töötlemiseks puudub, selle asemel seotakse iga juhtimiselemendi iga (töötlemist vajava) sündmusega nn väljakutsefunktsioon (callback-function). Seda tehakse käsuga *gtk_signal_connect()*, mille parameetriteks on juhtimiselemendi identifikaator, sündmuse tüüp ja



Joonis 2.6. Juhtprogrammi ekraanipilt Linux-keskkonnas

callback-funktsiooni aadress (lk xliv, xlv). Juhtimiselementide paigutus peaaknas on näha joonisel 2.6. Järgnevalt avatakse kommunikatsiooniks RS232-port käsuga *fdOpen()* ja käivitatakse programmi teatetsükkel käsuga *gtk_main()* (lk xlv). Teatetsükli realiseerib GDK-alamsüsteem.

Nupule "*Start*" vajutamisel kutsutakse välja selle nupu *callback*-funktsioon *StartStopProc()* (lk xli). Funktsiooni alguses kontrollib programm kasutaja poolt redigeerimislahtritesse sisestatud väärtusi. Sobimatu väärtuse puhul kirjutatakse peaakna staatusribale veateade, viiakse sisestusfookus vigase väärtusega lahtrile ja väljutakse funktsioonist. Kui kõik sisestatud väärtused on kontrollitud, arvutatakse graafiku kujutamise parameetrid ja muudetakse peaakna juhtimiselemendid mitteaktiivseteks käsuga *gtk_widget_set_sensitive()*. Nupu tekstiks seatakse funktsiooni *gtk_label_set_text()* abil "*Stopp*".

Suhtlemiseks RS232-pordiga luuakse Linux-programmis eraldi lõim (thread), nagu ka Windows-programmis (vt punkt 2.5.1), ainult siin on kõik mõõtmisprotseduurid koondatud kahe lõimefunktsiooni - *T66Proc0()* ja *T66Proc1()* - koosseisu. Uus lõim luuakse käsuga *Clone()* (lk xliii). Pogrammi töö juhtimine ja lõimedevaheline infovahetus on organiseeritud muutuja

g_Olukord ja alamfunktsiooni *KellProc()* abil (lk xxxix). See funktsioon registreeriti programmi alguses käsuga *gtk_timeout_add()* (lk xlv) ja teda kutsutakse ülejäänud programmist sõltumatult välja iga sekundi tagant. Funktsioon *KellProc()* kontrollib programmi teiste osade poolt seatud muutuja *g_Olukord* väärtust ja vastavalt sellele loob dialoogiaknaid, kirjutab teateid peaakna staatusreale, muudab juhtimiselementide omadusi ja kutsub esile spektrigraafiku ülejoonistamist, tekitades graafikuaknale sündmuse "*expose_event*".

T66Proc0() (lk xxxv) realiseerib kontrolleri initsialiseerimise ja reeperlüliti otsimise, nagu näidatud joonisel 2.4. Seejärel seab ta muutuja *g_Olukord* väärtuseks *3* ja lõpetab oma tegevuse. Mitte hiljem kui 1 sekundi pärast kutsutakse välja funktsioon *KellProc()*, mis loob uue modaalse dialoogiakna käsuga *gtk_dialog_new* (lk xxxix) ja sellele nupud *OK*, *Cancel* ning lahtri lainepikkuse väärtuse sisestamiseks. Nuppudega assotsieeritakse *callback*-funktsioonid *DlgOKProc()* (lk xxxiv) ja GTK-süsteemi vaikimisi-funktsioon *gtk_widget_destroy()*. Kui kasutaja vajutab nuppu "*OK*", kontrollitakse kõigepealt redigeerimislahtrisse sisestatud väärtust ja seejärel kutsutakse välja funktsioon *gtk_widget_destroy()*, mis omakorda kutsub välja funktsiooni *DlgQuitProc()* (lk xxxiv). See funktsioon seab muutuja *g_Olukord* väärtuseks *6*, mis põhjustab funktsiooni *KellProc()* järgmisel väljakutsel uue lõime loomise funktsiooniga *T66Proc1()* (lk xxxvi). *T66Proc1()* realiseerib ülejäänud osa mõõtmisest, st lähtekoordinaadile mineku ja spetri samm-sammulise skaneerimise (joonis 2.4). Iga mõõdetud spektripunkti järel kasvatab *T66Proc1()* muutuja *g_Punkt* väärtust. Kord sekundis kutsutakse välja funktsioon *KellProc()* (lk xxxix). Kui muutuja *g_Punkt* väärtus on kasvanud, tekitab *KellProc()* peaaknale sündmuse *expose_event*, mis põhjustab selle sündmusega assotsieeritud *callback*-funktsiooni *Protseduur2()* väljakutsumise (lk xxxiv). Selle funktsiooni sees toimub spektrigraafiku uuendamine GDK joonistamiskäskude abil.

Mõõtmise lõppemisel seatakse *T66Proc1()* poolt muutuja *g_Olukord* väärtuseks *11*. Selle peale *KellProc()* salvestab mõõtmistulemused kettafaili ja emuleerib nupule "*Stopp*" vajutamist funktsiooniga *gtk_signal_emit_by_name()*. Kutsutakse välja funktsioon *StartStopProc()* (lk xli), kus peaakna juhtimiselemendid viiakse olekusse, mis võimaldab uue mõõtmise parameetrite sisestamist. Nupu tekstiks seatakse funktsiooni *gtk_label_set_text()* abil "*Start*".

Kui lõimefunktsioon mõõtmise käigus tuvastab RS232-pordi sidekatkestuse, seab ta muutuja *g_Olukord* väärtuseks *12* ja lõpetab oma tegevuse. Selle peale *KellProc()* kutsub *gtk_signal_emit_by_name()* abil välja nupuga "*Stopp*" assotsieeritud funktsiooni *StartStopProc()* (lk xli). Sama funktsioon kutsutakse välja ka siis, kui kasutaja vajutab "*Stopp*"-nuppu. Funktsiooni *StartStopProc()* sees rakendatakse lõimefunktsiooni lõpetamiseks igaks juhuks

Kill()-käsku ja seejärel reinitsialiseeritakse kontrolleriplaat, saates RS232 DTR-liinile loogilise 1-impulsi käsuga *IOCtl()* (vt punkt 2.3).

Peaakna sulgemisele vastab sündmus *destroy*, millega on assotsieeritud *callback*-funktsioon *quit_app()* (lk xxxiv). Selle funktsiooni sees vabastatakse peaaknaga seotud ressursid käsuga *dispose()* ja lõpetatakse programmi elutsükkel käsuga *gtk_main_quit()*.

2.6 Linuxipõhine graafiline terminalisüsteem

Ehkki eelmises punktis kirjeldatud programm võib töötada iseseisvas Linux-operatsioonisüsteemiga arvutis, on mõõtmisskeem praktikumis otstarbekam realiseerida konfiguratsioonina, mida tuntakse kõvakettata terminali (diskless terminal system) nime all. See skeem sisaldab kahte arvutit: serverit ja terminali, mis on omavahel ühendatud arvutivõrgu kaudu. Terminalarvuti ei sisalda kettaseadmeid, alglaadimiseks vajalik info asub võrgukaardil EPROM-mälus. Pärast käivitamist laeb terminal serverarvutist operatsioonisüsteemi tuuma (kernel), monteerib failisüsteemi serverarvuti kõvakettale ja käivitab graafilise keskkonna (X-System). Edaspidi toimub programmide käivitamine terminalarvutis tavapärasel viisil. Lisaks monokromaatori juhtprogrammile on kasutada võrgubrauser, SSH-klient jm abivahendid. Selline süsteem omab mitmeid eeliseid: a) turvalisus - kõik programmid, katseandmed, konfiguratsiooniinfo jms asub serverarvutis, millele tavakasutajal ligipääsu pole; b) hind - terminalarvutiks sobib praktiliselt iga 486-klassi masin (operatiivmäluga vähemalt 16MB), nõutav on vaid võrgukaardi olemasolu; üks serverarvuti võib teenindada mitut terminali; c) kasutamismugavus - kuna terminalarvutil kettaseadmed puuduvad, ei tekita ta müra; samuti on terminali võimalik alati välja lülitada, kartmata failisüsteemi riknemist.

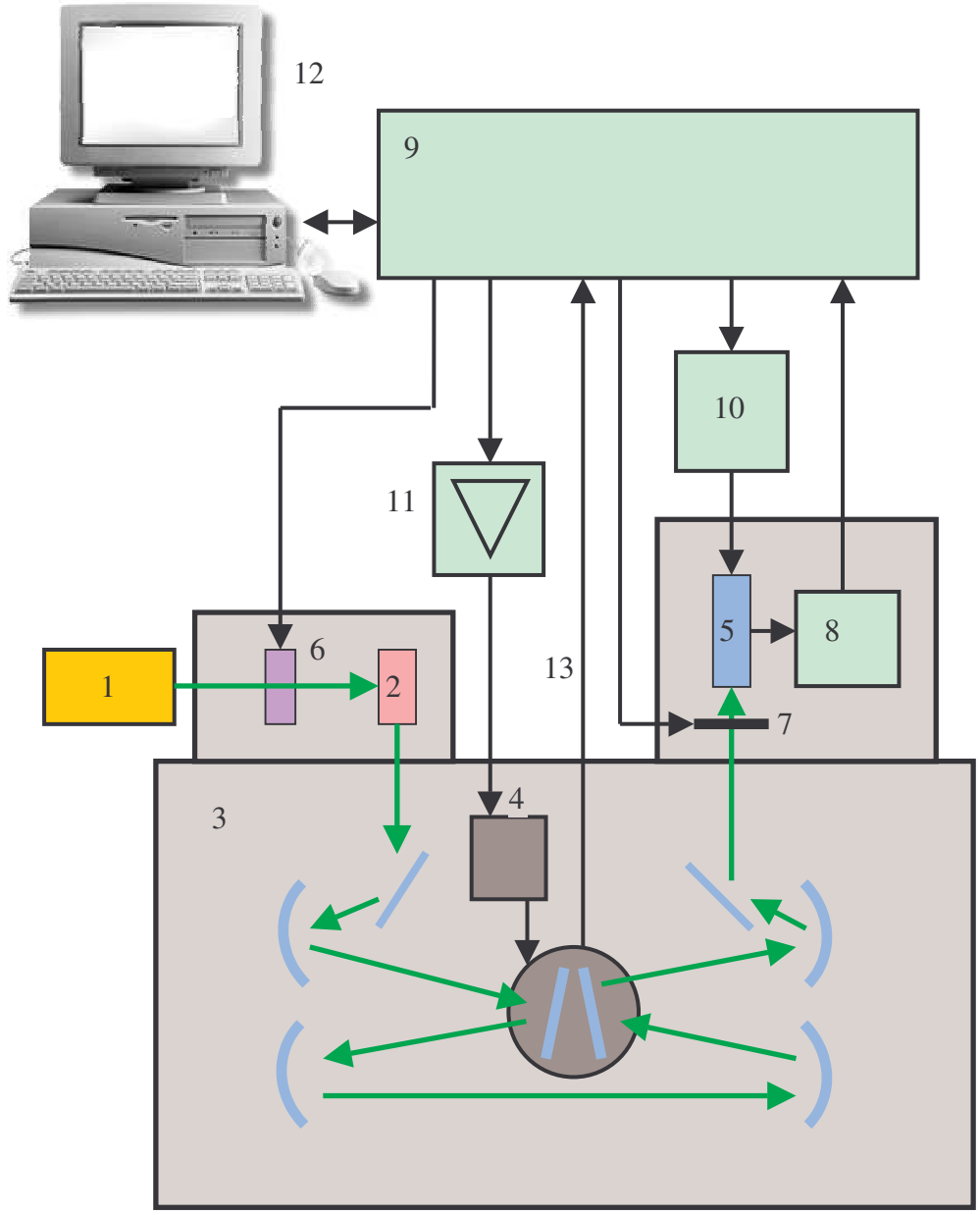
3 Mõõtekompleks kombinatsioonhajumisspektromeetriga DFS-52

3.1 Struktuur ja ülesanne

Spektromeeter DFS-52 on ette nähtud vedelike, kristallide ja pulbriliste ainete kombinatsioonhajumisspektrite registreerimiseks. Spektrite ergastamine toimub Nd:YAG laseri abil (λ=533 nm). Spektromeeter sisaldab kahekordset monokromaatorit, mille väljundis registreerib signaali üheelektroonses režiimis fotoelektronkordisti (FEK). Mõõtekompleksi kuuluvad veel toite- ja lisaplokid signaali töötlemiseks ja monokromaatori juhtimiseks, suurem osa neist plokkidest on käesoleva töö raames ümber ehitatud või asendatud. Välja on töötatud kontrolleriplokk seadme ühendamiseks arvutiga ja loodud tarkvara mõõtmisprotsessi juhtimiseks operatsioonisüsteemi MS Windows graafilisele kasutajaliidesele. DFS-52 baasil on formuleeritud praktikumitöö "Ramani hajumise uurimine".

3.2 Struktuuriskeem

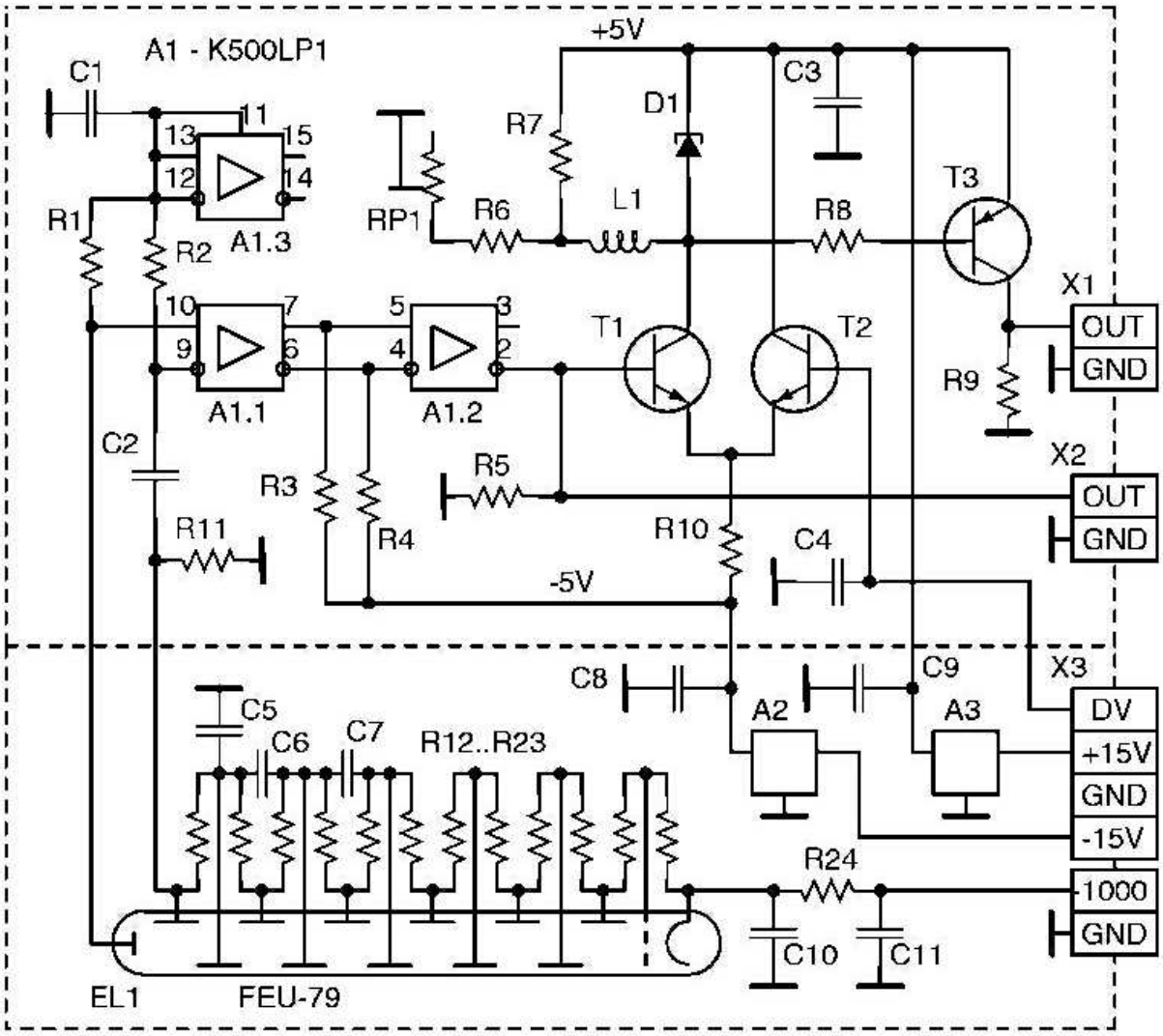
Mõõtekompleksi struktuuriskeem on toodud joonisel 3.1. Laser 1 valgustab küvetis 2 asuvat uuritavat vedelikku. Laserikiire intensiivsust on võimalik vähendada elektromehaanilise ajamiga valgusfiltriga 6. Küvetist hajunud valgus koondatakse kahekordse monokromaatori 3 esimesele pilule. Spektri skaneerimine toimub samm-mootoriga 4 varustatud mehhanismi abil. Monokromaatori väljundpilult satub uuritav kiirgus fotoelektronkordistile 5. Väljundpilu ja FEK vahel paikneb elektromehaaniline katik 7. Vastassuunalised pingeimpulsid fotoelektronkordisti anoodilt ja viimaselt dünoodilt antakse võimendi-diskriminaatori 8 sisendisse. Diskrimineeritud üheelektroonsed impulsid juhitakse kontrolleriplaadile 9. FEK toitepinge saabub plokist 10, mille väljundpinget reguleeritakse kontrolleriplaadilt saabuvate signaalidega. Toitepinge on reguleeritav vahemikus 0..-2200V. Monokromaatori difraktsioonvõresid pööratakse samm-mootori abil, mille juhtpinged saabuvad läbi võimendite 11 kontrolleriplaadilt. Monokromaatori skaneerimismehhanismi reeperlüliti signaal suundub samuti kontrolleriplokki. Ülejäänud aparatuuri moodustavad toiteplokid monokromaatori ahelate tarvis ning juhtarvuti 12. Spektromeetris DFS-52 on ette nähtud võimalus mõõta küvetile langeva laserikiirguse intensiivsust teise fotoelektronkordisti abil (nn võrdluskanal). Ka see FEK töötab üheelektroonses režiimis ja omab täpselt samasugust diskriminaator-võimendit, kui esimene (nn otsekanali) FEK. Võrdluskanal ei ole praegu kasutusel ja teda ei ole plokkskeemil 3.1 näidatud.



Joonis 3.1. Spektromeetriga DFS-52 mõõtekompleksi plokkskeem.
1 - laser, 2 - küvett uuritava vedelikuga, 3 - monokromaator, 4 - samm-mootor, 5 - fotoelektronkordisti (FEK), 6 - nõrgendusfilter, 7 - katik, 8 - võimendi-diskriminaator, 9 - kontrolleriplaat, 10 - FEK toiteplokk, 11 - samm-mootori võimendiplokk, 12 - juhtarvuti, 13 - signaal reeperlülitilt

Kontrolleriplokil 9 on aga olemas sisend võrdluskanali FEK impulsside loendamiseks (vt jaotis 3.4). Võrdluskanali kasutuselevõtuks on vaja ehitada veel üks võimendi-diskriminaatoriplokk (jaotis 3.3) ja muuta juhtprogramme.

Mõõtmisprotsessi kontrollib juhtarvuti programm koostöös kontrolleriplaadi protsessoriga. Side kontrolleriplaadi ja juhtarvuti vahel toimub RS232-protokolli abil. Monokromaatori DFS-52 parameetrid ja mõõtekompleks on põhjalikumalt kirjeldatud mujal [16]. Käesoleva töö käigus on ümber ehitatud fotoelektronkordisti võimendi-diskriminaatoriplokk 8, lisatud



Joonis 3.2. Võimendi-diskriminaatori põhimõtteskeem

elektromehaanilised ajamid nõrgendusfiltrile 6 ja katikule 7 ning kirjutatud juhtprogrammid. Realiseeritud on ka fotoelektronkordisti üheelektroonsete karakteristikute mõõtmise automaatne skeem.

3.3 Võimendi-diskriminaatoriplokk.

Ploki ehitamisel on võetud eeskujuks kirjanduses [12..14] ilmunud skeemid. Võimendi-diskriminaatori põhimõtteskeem on kujutatud joonisel 3.2. Vastandfaasis pingeimpulsid fotoelektronkordisti anoodilt ja viimaselt dünoodilt saabuvad võimendi A1.1 diferentsiaalsisenditele 10 ja 9. Mikroskeem K500LP1 kujutab endast emittersidestusloogika (ESL) siinivõimendit, mida laialdaselt kasutatakse ka kõrgsageduslike signaalide võimendamiseks [12..14]. Mikroskeem sisaldab kolme ühesugust võimendit, millest siin on kasutusel kaks. Võimendi viiakse lineaarsesse töörežiimi mikroskeemi 11. viigult saabuva

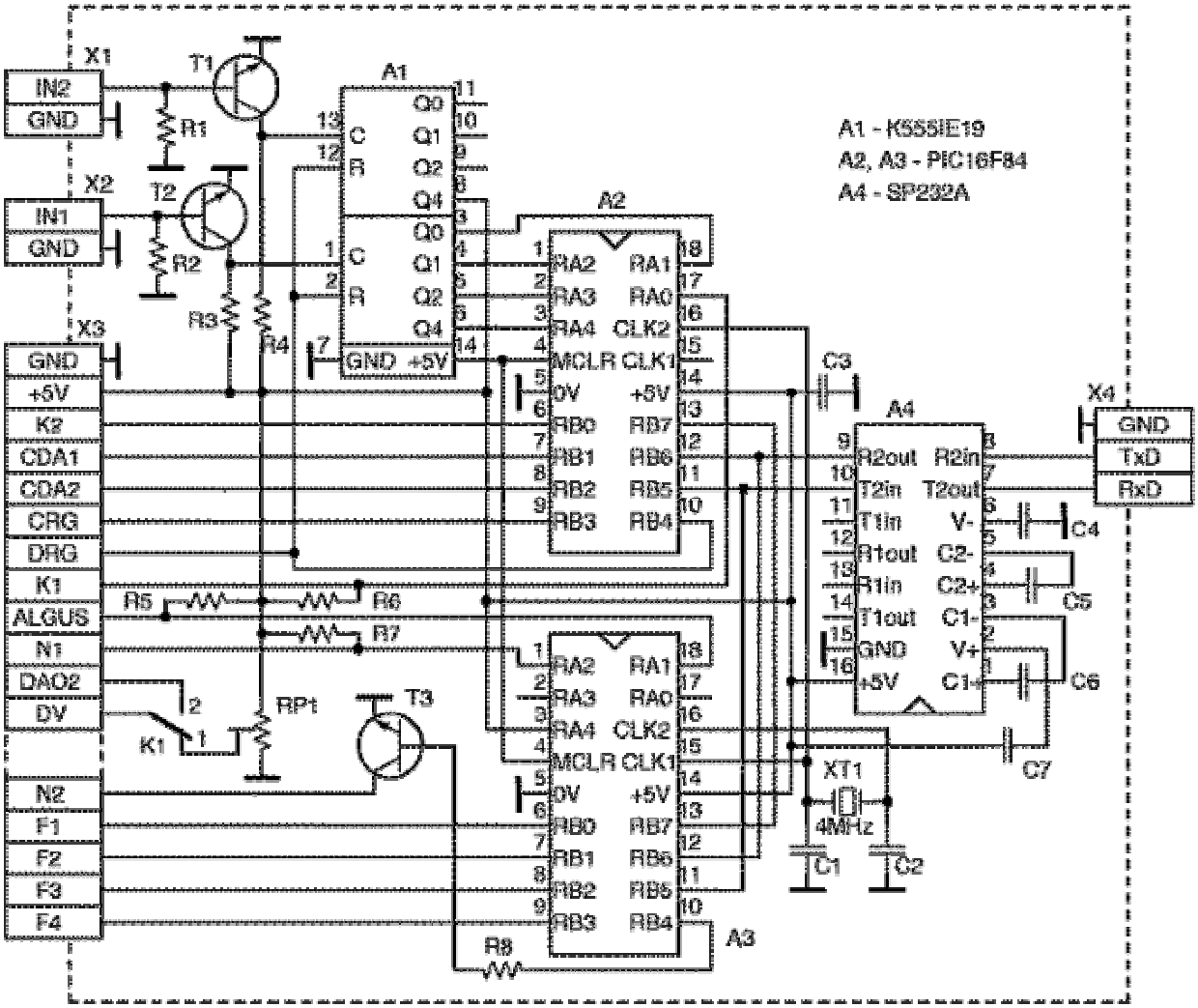
tugipinge abil, mis juhitakse läbi takstite R1, R2 võimendi sisenditele. R1 on ühtlasi FEK anoodkoormustakistiks. Esimese võimendusastme väljunditelt 6, 7 juhitakse signaal vastandfaasis teise võimendusastme A1.2 sisenditele 4, 5. Võimendatud (positiivsed) impulsid juhitakse A1.2 väljundilt 2 diskrimineerimislülitusse, mis põhineb transistoridel T1, T2 ja tunneldioodil D1. Signaal võimendi väljundist juhitakse ka ploki välisküljel asuvasse pessa X2.

Diskriminaatori tööpõhimõte on järgmine. Potentsiomeetri RP1 ja transistori T2 baasipinge abil on diferentsiaalaste transistoridel T1, T2 viidud sellisesse režiimi, et positiivse impulsi puudumisel T1 baasil asub tunneldioodi tööpunkt pinge-voolutunnusjoone esimesel, tõusval harul. Transistori T1 kollektorivool hargneb läbi dioodi D1 ja ahela L1, R7. Positiivse pinge saabumisel T1 baasile hakkab tema kollektorivool suurenema. Alates kollektorivoolu teatud väärtusest satub tunneldiood tunnusjoone langevale, negatiivse diferentsiaaltakistusega lõigule. Areneb laviinitaoline protsess, mille käigus pinge dioodil kasvab (ja T1 kollektoril langeb), kuni dioodi töörežiim on jõudnud tunnusjoone kolmandale, tõusvale harule. Selle käigus saab ühtlasi võimendustransistor T3 avava baasivoolu impulsi ja tema kollektoritakistil R9 tekib positiivne pingeimpulss. Ümberlülitumise hetk (ja seega diskrimineerimisnivoo T1 baasile saabuva impulsi suhtes) on määratud T2 baasil hoitava alalispingega. Impulsi pikkus on määratud peamiselt induktiivsuse L1 väärtusega ja on valitud umbes 30 ns . Laeng hakkab dioodi elektroodidelt ära voolama L1, ja R7 kaudu, kuni temast enam ei piisa D1 hoidmiseks tunnusjoone kolmandal harul ja lülitus naaseb lähteolekusse.

Konstruktiivselt on võimendi-diskriminaatoriplokk kujundatud punasest vasest karbina, mis kinnitub jäigalt fotoelektronkordisti pesa külge. Ploki eraldatud sektsioonis asuvad FEK dünoodpingejaguri takistid, integraalstabilisaatorid (tüübiga 7805, 7905) ±5V toitepinge tekitamiseks ja ühenduspesad. Plokki saabuvad järgmised pinged: kõrgepingeplokist 10 (joon. 3.1) FEK toitepinge; kontrolleriplokist 9 diskrimineerimispinge transistori T2 baasile; toitepinged ±15V. Diskrimineeritud impulsid juhitakse võimendustransistori T3 kollektorilt pistiku X1 ja koaksiaaljuhtme kaudu kontrolleriplaadile 9 (joonis 3.1).

3.4 Kontrolleriplokk

Kontrolleriplokk juhib spektromeetri erinevate sõlmede tööd ja organiseerib andmevahetust juhtarvutiga, tema põhimõtteskeem on toodud joonisel 3.3. Ploki koosseisu kuulub neli mikroskeemi: kahekanaliline loendur/sagedusjagur A1 (K555IE19), mikrokontrollerid A2, A3 (PIC16F84) ja RS232-siini draiver A4 (SP232A). Signaalid PIC-kontrollerite A2, A3 viikudel on toodud tabelites 3.1 ja 3.2.



Joonis 3.3. Kontrolleriploki põhimõtteskeem

Impulsid otse- ja võrdluskanali diskriminaator-võimendite väljundeist saabuvad A1 kummagi kanali loendussisendeile. Mõõtmistsükli alguses nullitakse loendurid signaali R abil, mis saabub A2 viigult RB4. Loendurite 4-bitised väljundkoodid juhitakse PIC-kontrollerite sisenditele, kusjuures vanim bitt on ühtlasi PIC sisemise 8-bitise aparaatse loenduri TMR0 sisendsignaaliks (vt lisa II). Selle loenduri ületäitumisel (st 257. biti saabumisel PIC viigule RA4) toimub kontrolleri sees programmiline ülekanne kõrgemasse baiti. Otsekanali impulsiloenduri summaarne bittide arv on seega 4+8+8=20, mis tagab registreerimisel piisava dünaamikaulatuse. Võrdluskanalis on kasutusel vaid A1 vanim väljundbitt Q4 ja loenduri summaarne bittide arv seega 16. Välise aparaatse loenduri kasutamine võimaldab vähendada impulsside sagedust PIC-kontrolleri loendussisendil ja vähendada programmilise ülekandega seotud "surnud aja" mõju, eriti arvestades viivitusi katkestuste teenindamisel. Samal põhjusel, ja samuti mõõtmisskeemi sümmeetrilisuse huvides, kasutatakse otse- ja võrdluskanali impulsside loendamiseks eraldi PIC-kontrollereid. Kahe kontrolleri kasutamine võimaldab ka lihtsustada ploki skeemi, kuna suure

viigu nr	nimetus	signaal	kirjeldus
1	RA2	Q1	loenduri A1 väljundsignaal
2	RA3	Q2	"
3	RA4	Q4	"
6	RB0	K2	FEK katiku ajam
7	RB1	CDA1	DA-muunduri juhtimine
8	RB2	CDA2	"
9	RB3	CRG	"
10	RB4	DRG, R	", loendurite A1 nullimine
11	RB5	RxD	RS232 signaal kontrollerilt arvutile
12	RB6	TxD	RS232 signaal arvutilt kontrollerile
13	RB7	-	PIC-de sünkroniseerimine
17	RA0	K1	katiku ajami kontroll-lüliti
18	RA1	Q0	loenduri A1 väljundsignaal

Tabel 3.1. Signaalid kontrolleri A2 viikudel

arvu programmeeritavate väratite olemasolu kaotab vajaduse täiendavate hoideregistrite ja multipleksorite järele. Et PIC-kontrollerid töötavad ühise sisendiga (signaalid TxD juhtarvutilt A2, A3 viikudele 11) ja väljundiga (signaalid RxD arvutile viikudelt 12), siis on vajalik nende omavaheline sünkroniseerimine. Sel otstarbel on PIC-d ühendatud oma 13.-te viikude kaudu. PIC-d töötavad ühise taktgeneraatoriga sagedusel 4 MHz.

Kõik A2 ja A3 poolt teostatavad operatsioonid on ajaliselt eraldatud, v. a. impulsside loendamine ja instruktsiooni ootamine juhtarvutilt. RS232-siiniga suhtlemise protokoll on järgmine. Arvuti jadaport peab olema konfigureeritud andmeedastuskiirusele 9600 baudi, andmesõnas 8 bitti, 1 stopp-bitt, 0 paarsusbitti. Kõik arvutipoolsed instruktsioonid on 1-baidised, mõni sisaldab ka ühe- või kahebaidilist argumenti. Käsud on PIC-de vahel ära jagatud. Pärast toite sisselülitamist siirduvad PIC-kontrollerid režiimi, milles ootavad instruktsiooni juhtarvutilt. RS232-signaali töödeldakse programmiliselt ja see toimub mõlemas PIC-s sarnaselt. Kehtiva instruktsiooni

viigu nr	nimetus	signaal	kirjeldus
1	RA2	N1	nõrgendusfiltri kontroll-lüliti
2	RA3	-	-
3	RA4	Q4	loenduri A1 väljundsignaal
6	RB0	F1	faasipinged samm-mootorile
7	RB1	F2	"
8	RB2	F3	"
9	RB3	F4	"
10	RB4	N2	nõrgendusfiltri ajam
11	RB5	RxD	RS232 signaal kontrollerilt arvutile
12	RB6	TxD	RS232 signaal arvutilt kontrollerile
13	RB7	-	PIC-de sünkroniseerimine
17	RA0	-	-
18	RA1	ALGUS	signaal monokromaatori reeperlülitilt

Tabel 3.2. Signaalid kontrolleri A3 viikudel

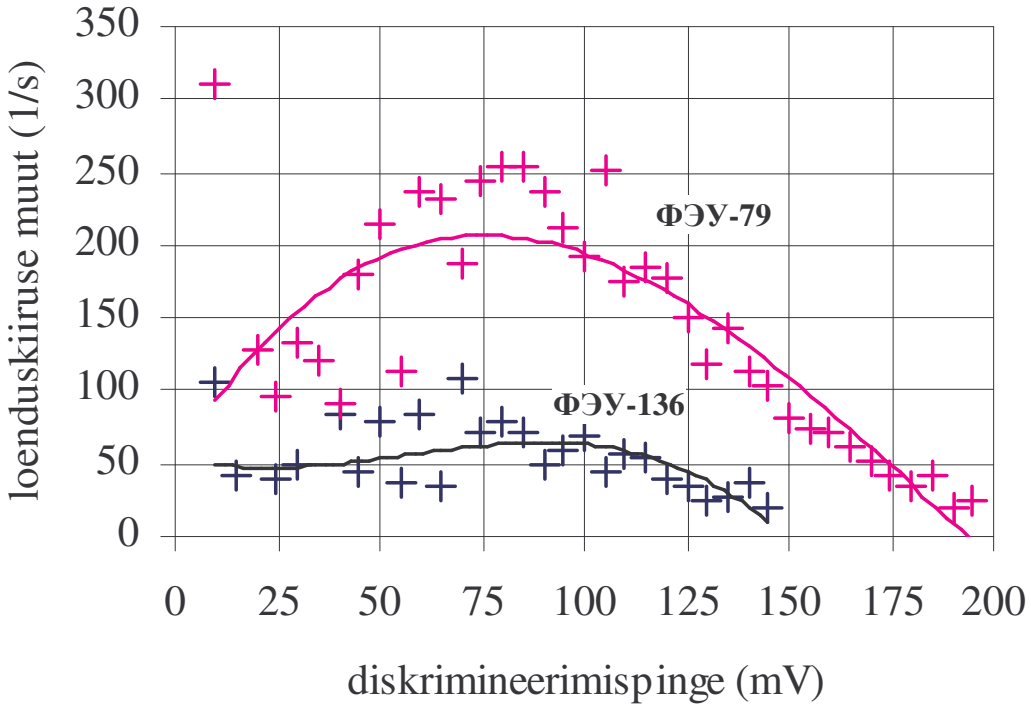
käsubait	argumendid	kontrolleri vastus	kirjeldus
"K"	loendusaeg (1b)	0x00	loendusaja seadmine
"S"	-	"	skaneerimismehhanism lähteseisu
"m"	sammude arv (2b)	"	liikuda lainearvu kasvu suunas
"M"	"	"	liikuda lainearvu kahanemise suunas
"I"	-	"	nõrgendusfilter ette
"P"	-	"	nõrgendusfilter eest ära
"i"	-	"	sulgeda FEK katik
"p"	-	"	avada FEK katik
"f"	-	"	kõrvaldada mootori faasipinged
"A"	-	loendustulemus (5b), 0x00	loendada impulsse
"D"	pinge (2b)	"	seada DA I kanali väljundpinge
"d"	"	"	seada DA II kanali väljundpinge

Tabel 3.3. Kontrollerite A2, A3 juhtkäsud

dešifreerimisel "peegeldab" vastav PIC arvutile sama baidi tagasi ja seab oma väljundile RB7 (13. viik) loogilise 1-nivoo. Seda nivood kontrollib teine PIC enne iga siinilt saabuva baidi tõlgendamist. See võimaldab ära hoida vigu, mis tekivad, kui ühele PIC-le saadetud käsu argumenti võiks teine tõlgendada instruktsioonina. Kui käsuga käib kaasas argument, saadab arvuti selle kontrollerile niipea, kui on registreerinud "peegeldatud" baidi. Käsu täitmisest informeerib PIC juhtarvutit baidi 0x00 saatmisega. Samuti viib ta oma väljundi B7 (13. viik) kõrge impedantsiga olekusse teavitamaks teist PIC-i, et järgmine arvutilt saabuv bait kuulub tõlgendamisele kui võimalik instruktsioon.

FEK impulsside loendamise protseduur on mõlemale PIC-le ühine ja nõuab nende täiendavat faseerimist. A3 vastava instruktsiooni baiti arvutile tagasi ei saada, et vältida konflikti väljundil (A2, A3 kokkuühendatud viigud 11). Impulsside loendamine toimub A2-s ja A3-s erinevalt. A2 loendab alates instruktsiooni saabumisest aega taktgeneraatori abil, kogudes impulsse viigult RA4 sisemisse 8-bitilisse aparaatsesse loendurisse (programmilise ülekandega kõrgemasse baiti). Kui mõõtmiseks määratud intervall lõpeb, loeb A2 tulemuse noorimad bitid A1 väljunditelt Q0..Q4 ja püstitab oma 13. viigul loogilise 1-nivoo. Kontrolleris A3-s sisaldab loendamistsükkel viigu 13 kontrollimist. Kõrge nivoo avastamisel loendamine lõpetatakse ja A3 siirdub ooterežiimi. Loendamise tulemuse (20 bitti) saadab arvutile kõigepealt A2. Seejärel kõrvaldab ta loogilise 1-nivoo oma 13.-lt viigult, mis on märgiks A3-le, et liin on saatmiseks vaba. A3 saadab arvutile võrdluskanali loendustulemuse (16 bitti) ja tsükli lõpu tunnusena baidi 0x00. Seejärel on kontrollerid valmis järgmiseks instruktsiooniks.

Instruktsioonide loetelu on toodud tabelis 3.3 ja PIC-kontrollerite programmid, mis neid instruktsioone realiseerivad, lisas II.



Joonis 3.4. Fotoelektronkordistite diskrimineerimistunnusjooned

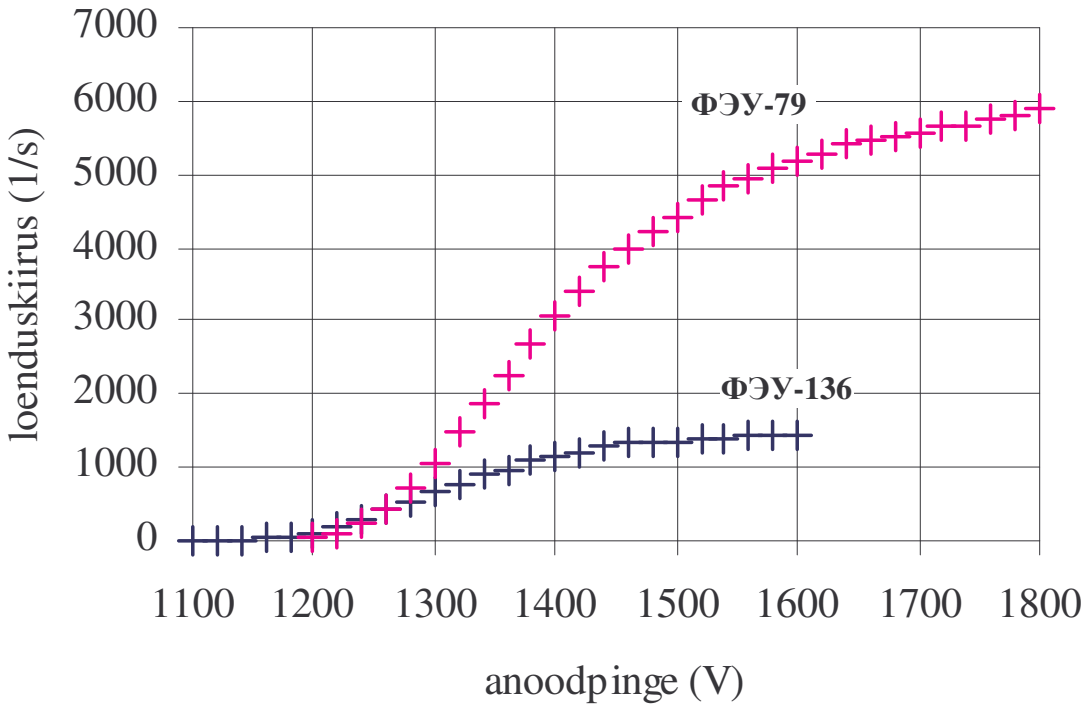
PIC-kontrollerite kasutamise kohta on võimalik leida informatsiooni tootjafirma koduleheküljelt [5].

3.5 Üheelektroonsete karakteristikute mõõtmise automaatskeem

Fotoelektronkordisti väljundimpulsside diskrimineerimise lävipinge saabub transistori T2 baasile (joon. 3.2) kontrolleriplaadi lülitilt K1 (joon. 3.3). Lüliti asendis "1" võetakse diskrimineerimispinge seadepotentsiomeetrilt RP1. Lüliti asendis "2" saabub diskrimineerimispinge kõrgepingeploki 10 (joon 3.1) DA-muunduri vaba kanali väljundist (signaal DAO2 joonisel 3.3). DA-muunduri väljundpinget seab PIC-kontroller A2 signaalide CRG, DRG, CDA1, CDA2 abil (joon. 3.3). Programmiliselt on võimalik muuta nii FEK anoodpinget kui ka väljundimpulsside diskrimineerimisnivood ja registreerida seega FEK anood- ja üheelektroonseid tunnusjooni. Näited mõlemast tunnusjoonest kahe FEK eksemplari jaoks on toodud joonistel 3.4 ja 3.5.

3.6 Juhtprogramm

Windowsi GUI-keskkonnas töötava juhtprogrammi täielik tekst on toodud lisas II. Programm on kompileeritav Borland Delphi või vabavaralise Free Pascali kompilaatoriga. Windowsi GUI-



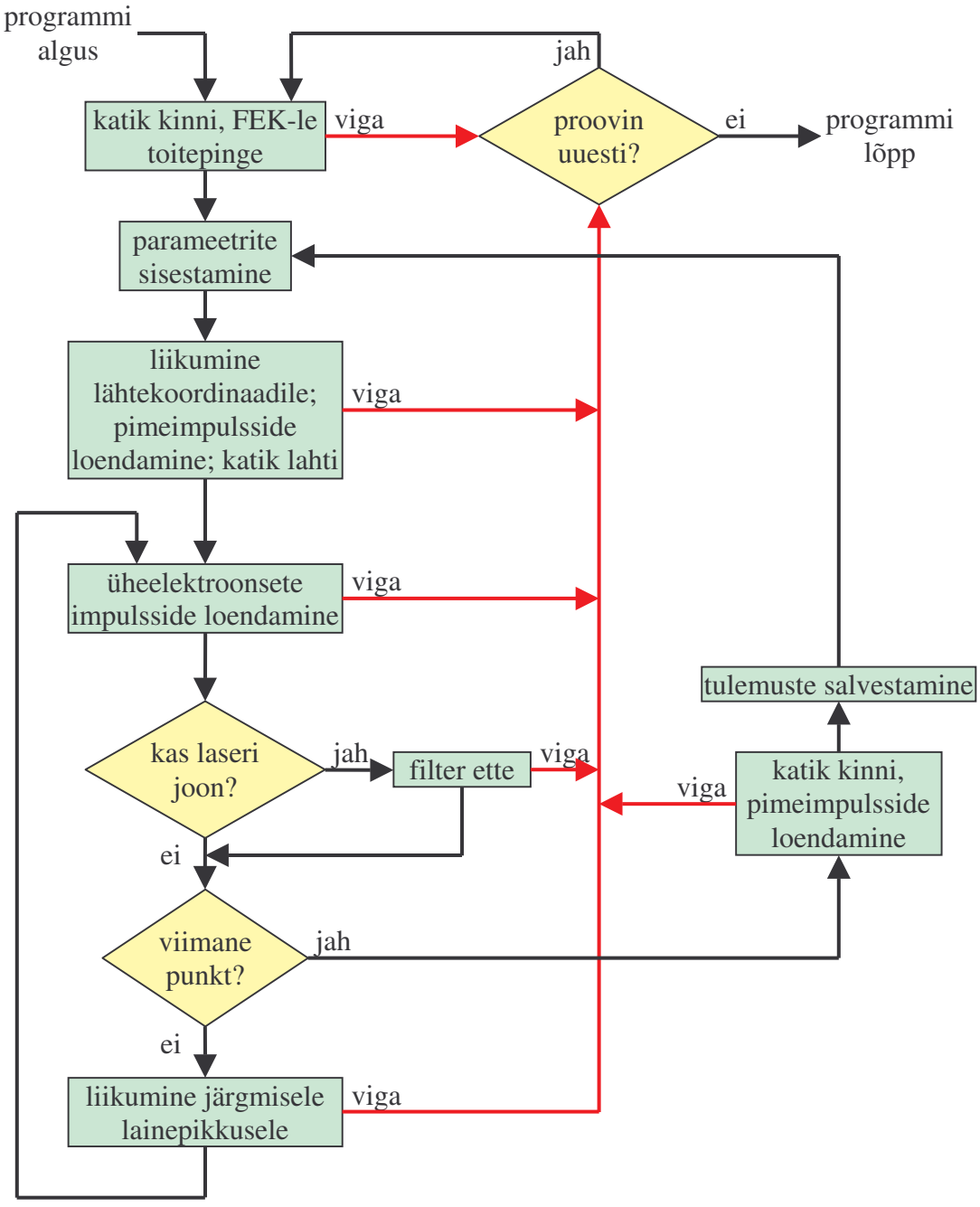
Joonis 3.5. Fotoelektronkordistite anoodtunnusjooned

programmi üldine struktuuriskeem ja peamiste programmeerimisalaste mõistete selgitused on toodud käesoleva töö lisas IV ja ühe konkreetse programmi põhjalikum kirjeldus punktis 2.5.1. DFS-52 juhtprogramm erineb eelmistest selle poolest, et programmi peamiseks aknaks on modaalne dialoogiaken (luuakse käsuga *DialogBox()*, lisa lk lxxiv). See lihtsustab programmi struktuuri: dialoogiakna käsutsükli (Message Loop) organiseerib operatsioonisüsteem automaatselt, st pole vaja kasutada API-funktsioone *GetMessage()*, *DispatchMessage()*. Samuti pole dialoogiakna jaoks vaja kutsuda välja funktsiooni *DefWindowProc()* (õigemini küll *DefDlgProc()*), kuna ka seda teeb operatsioonisüsteem automaatselt. Dialoogiaknal puudub menüü, aga muus osas on programmi struktuur sarnane punktis 2.5.1 kirjeldatule. Dialoogiakna ressursifaili lähtekood (nuppude, tekstilahtrite jm elementide paigutus) asub failis m95.rc (lisa II).

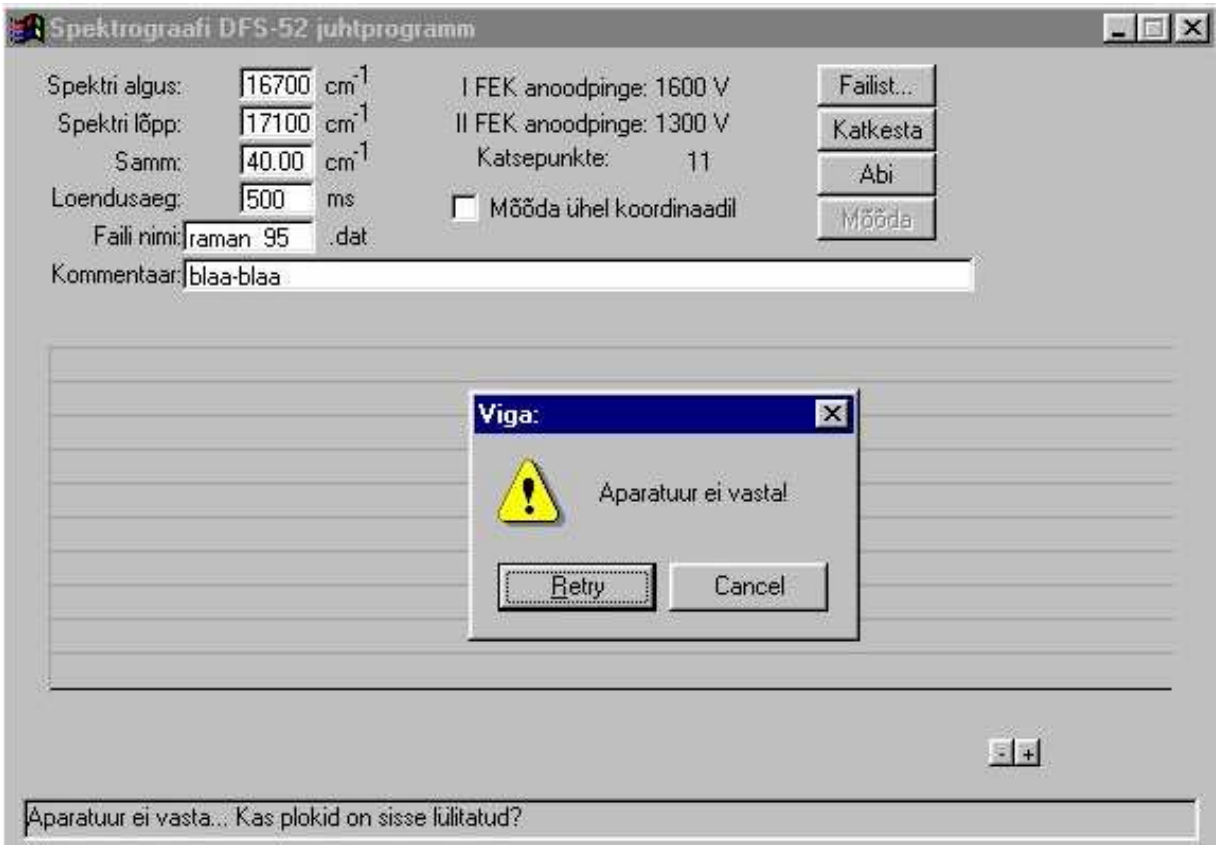
Peamise akna aknafunktsioon on *MainProc()* (lk lxvi). Esimene teade, mille operatsioonisüsteem aknafunktsioonile saadab, on *WM_INITDIALOG* (lk lxxiii). Selle teate töötlemise käigus toimub mõõtmisparameetrite vaikimisi-väärtuste lugemine failist raman_95.ini (vt lisa II) ja programm juhitakse *PostMessage()* abil järgmiste initsialiseerimisprotseduuride juurde (*WM_COMMAND*, *wParam*=601, lk lxxiii). Praktika näitab, et fotoelektronkordisti vajab üheelektroonse töörežiimi stabiliseerumiseks aega, sellepärast avatakse kohe programmi alguses RS232-port ja luuakse lõim funktsiooniga *T66Proc0()* (lk lix). Selle funktsiooni ülesandeks on sulgeda FEK ees asuv

katik 7 (joonis 2.1) ja lülitatada sisse FEK toitepinge (mille väärtus on eelnevalt loetud ini-failist). Kui see ei õnnestu, postitab lõim peaaknale teate *WM_COMMAND*, *wParam*=501 (lk lxx), mille peale viimane kuvab veateate ja loob uuesti lõime funktsiooniga *T66Proc0()*. Juhtprogrammi käivitamine väljalülitatud või mittetöötava aparatuuriga pole ette nähtud.

Järgnevalt on kasutajal võimalik sisestada mõõtmise parameetrid. Kasutada on nupud "*Mõõda*", "*Abi*" ja "*Välju*" (joonis 3.6). Nupu "*Mõõda*" valimisele järgneb sisestatud parameetrite kontroll (koos võimaliku veateatega) ja mõõtmisprotsess uute lõimede moodustamisega. Selle tegevuse



Joonis 3.5. Spektri skaneerimise plokkskeem



Joonis 3.6. Juhtprogrammi ekraanipilt

käik on põhjalikumalt kirjeldatud punktis 2.5.1. Joonisel 3.5 on esitatud operatsioonide loogiline järjestus kombinatsioonhajumisspektri registreerimisel, nagu ta on realiseeritud lõimefunktsioonide *T66Proc0()..T66Proc4()* abil. Programmi peamine aken funktsioneerib mõõtmisprotsessi suhtes asünkroonselt, st kasutajal on võimalik mõõtmine ükskõik millises faasis lõpetada, vajutades nuppu "*Katkesta*". Enne ja pärast spektri skaneerimist suletakse FEK ees katik 7 (joonis 3.1) ja loendatakse pimeimpulsse. Skaneerimisel ergastusjoone lainepikkuste läheduses nihutatakse laserikiire teele nõrgendav filter 6 (joonis 3.1). Lainearvud, mille vahel filter peab olema ette nihutatud, leitakse programmi käivitamisel ini-failist (sõltuvad kasutatavast laserist). Skaneerimisprotsessi lõppedes jääb katik 7 suletuks, FEK toitepinget välja ei lülitata. Katsetulemused salvestatakse ASCII-tekstifaili, mille formaat nähtub lisast II. Dialoogiakna juhtelemendid viiakse olekusse, mis võimaldavad uue mõõtmise parameetrite sisestamist.

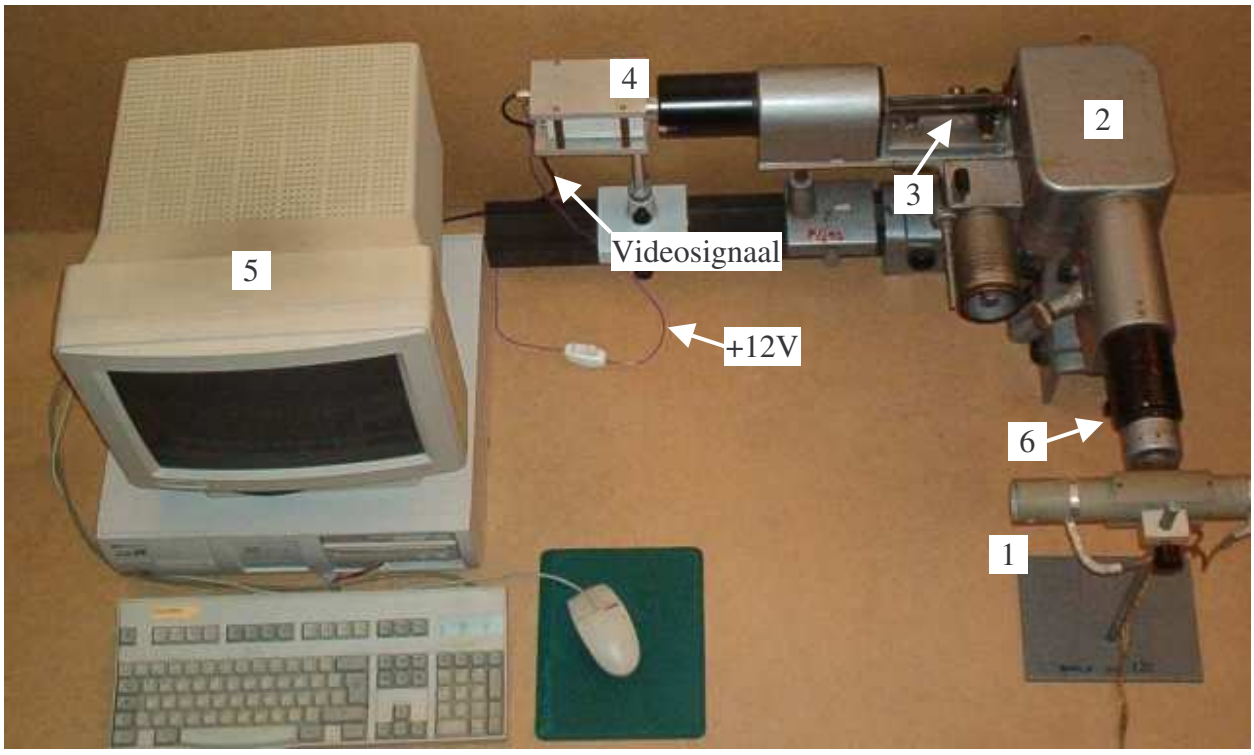
4 Mõõtekompleks spektrijoone ülipeenstruktuuri uurimiseks

4.1 Lähteülesanne.

Kaks praktikumitööd - "Fabry-Perot etaloni uurimine" ja "Lummeri-Gehrcke plaadi uurimine" - on aparaatselt ja programmiliselt ülesehituselt sarnased ja seepärast käsitletakse neid siin koos. Seadme aluseks on ümberehitatud prismamonokromaator UM-2, mille väljundobjektiivi ette on paigutatud interferomeeter. Interferentspilti registreerib videokaamera, mille signaal juhitakse arvutisse. Windowsi-põhine juhtprogramm võimaldab interferogrammi jälgimist ekraanil ja salvestamist.

4.2 Struktuurskeem

Aparatuuri ehitus selgub jooniselt 4.1. Elavhõbelamp 1 asetseb otse monokromaatori 2 sisendpilu ees. Monokromaatori prisma ja väljundobjektiivi vahele on paigutatud interferomeeter



Joonis 4.1. Praktikumitööde "Fabry-Perot' etaloni uurimine" ja "Lummeri-Gehrcke plaadi uurimine" aparaatosa
1 - elavhõbelamp, 2 - monokromaator, 3 - interferomeeter, 4 - videokaamera, 5 - juhtarvuti, 6 - monokromaatori katiku ajam

3 (Fabry-Perot' etalon või Lummeri-Gehrcke plaat). Selleks on prisma ja objektiivi vahekaugust suurendatud. Interferomeetrit ümbritseb kate, mis takistab hajunud valguse pääsemist monokromaatorisse. Väljundobjektiivi fokaaltasandis tekkivat interferogrammi registreerib videokaamera 4. Kaamera väljundsignaal juhitakse TV-kaardi kaudu arvutisse 5. Pimesignaali registreerimisel suletakse käsitsi monokromaatori katik 6. Kompleksi juurde kuulub veel elavhõbelambi reguleeritav toiteplokk; videokaamera saab toitepinge (+12V) juhtarvutist.

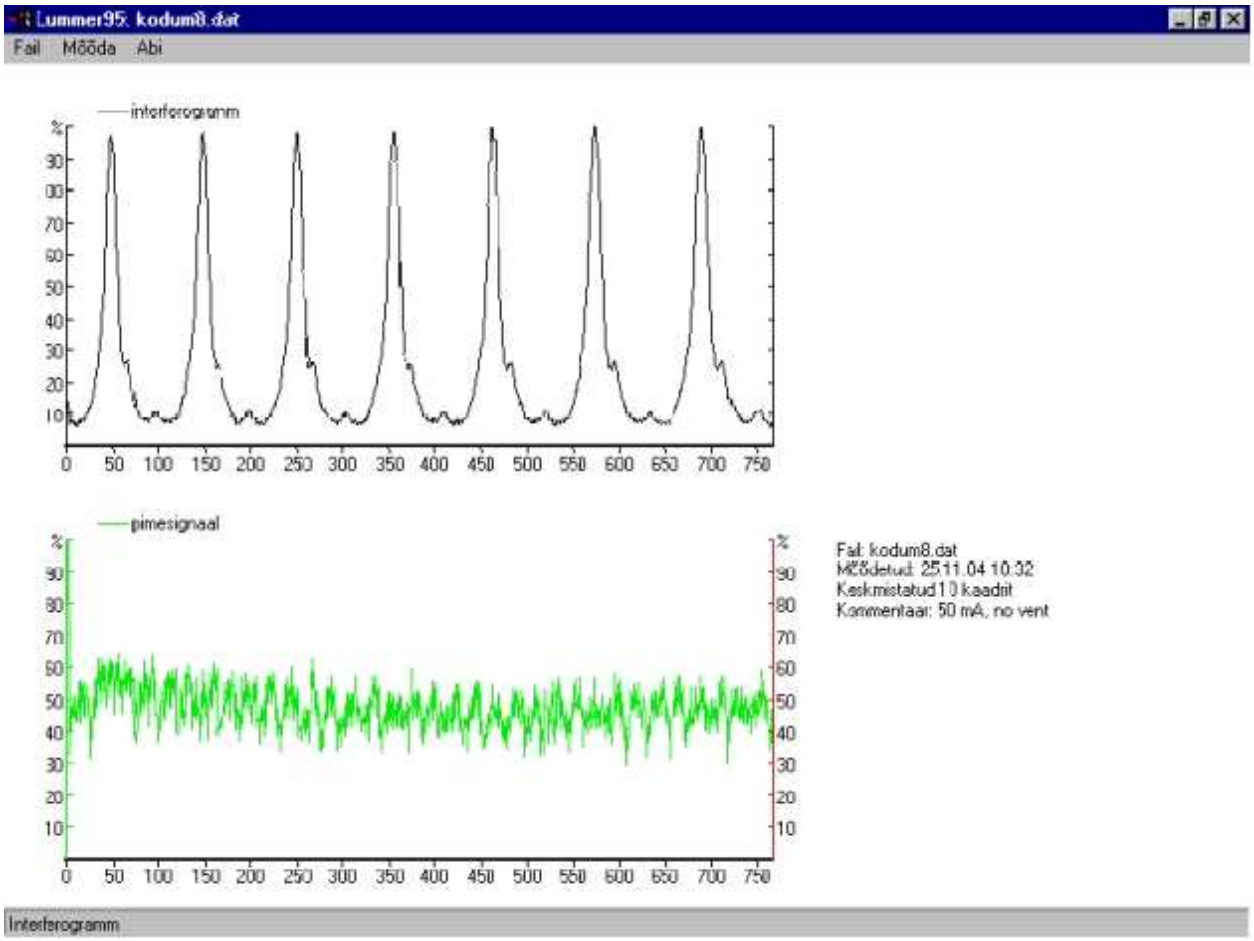
4.3 Juhtprogrammi lühikirjeldus

Programmi lähtetekst on toodud lisas III, kompileeritav on ta nii Borland Delphi keskkonnas kui ka vabavaralise kompilaatoriga Free Pascal. Windows-keskkonna GUI-programmi põhimõtteline struktuur ja mõistete selgitused on toodud lisas IV. Järgnevas kirjelduses on programmiteksti alamfunktsioonide ja muutujate nimed kujutatud kaldkirjas.

Videoakna loomine, kaadri salvestamine ja TV-kaardi parameetrite seadmine toimub alamsüsteemi "Video for Windows" (lühendatult “VfW”) [6] ja TV-kaardi draiverite abil. Selleks kasutatakse süsteemifailides avicap32.dll ja btvid32.ddl asuvaid funktsioone, mis lingitakse *external*-klausli abil (lk lxxviii, lxxix).

Programmi alguses registreeritakse peaakna klass käsuga *RegisterClass()* ja korraldatakse teadete töötlemise tsükkel (Message Loop) funktsioonidega *GetMessage()*, *DispatchMessage()* (lk xci). Peaakna aknafunktsiooniks on *MainProc()* (lk lxxxi). Esimene teade (Message), mille operatsioonisüsteem aknafunktsioonile saadab, on *WM_CREATE* (lk lxxxix). Selle teate töötlemise käigus loetakse ini-failist eksperimendi vaikimisi-parameetrid. ini-faili tekst on toodud lisas III. Interferentspildi mõõtmisel kasutatakse kaamerapildi keskelt kitsast horisontaalset piirkonda, millele tekib monokromaatori sisendpilu kujutis. Selle piirkonna algusele ja lõpule (skaneerimisridade mõttes) vastavad ini-failis konstandid *RIDA1* ja *RIDA2*. API-funktsiooniga *GetSystemMetrics()* leitakse ekraani geomeetrilised mõõtmed. Järgnevalt arvutuatakse välja mitmesugused abikonstandid videoakna ja juhtimiselementide paigutuse jaoks peaaknas (lk lxxxix). Peaakna menüü kirjeldus asub ressursiskriptis fabry.rc (vt lisa III).

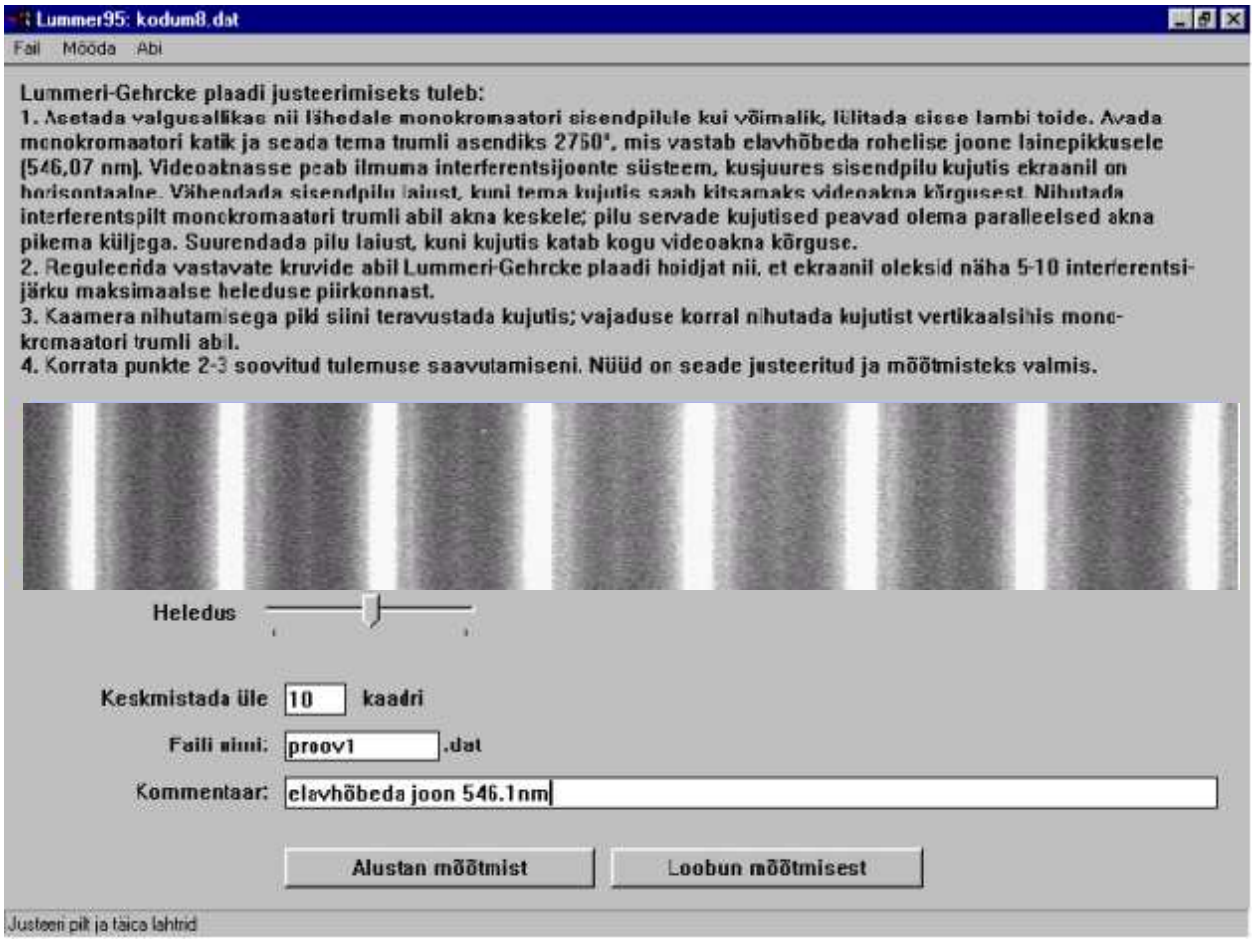
Menüüvaliku *Fail→Ava* korral loetakse sisse varasema mõõtmise andmefail (lk lxxx). Mõõtmistulemusi hoitakse ASCII-tekstifailis, mille struktuur on toodud lisas lk xcii. Interferogramm ja pimesignaal joonistatakse graafikuna peaakna pinnale. API joonistamisfunktsioone, nagu GUI-programmis ikka, võib välja kutsuda ainult funktsioonide *BeginPaint()* ja *EndPaint()* vahel teate *WM_PAINT* töötlemise käigus (lk lxxxi). Programmi töötamise jooksul joonistatakse peaaknasse erinevaid objekte. Kuna teade *WM_PAINT* võidakse



Joonis 4.2. Juhtprogrammi ekraanipilt faili avamisel

operatsioonisüsteemi poolt aknafunktsioonile postitada suvalisel ajal, peab alati olema teada, mida aknale joonistada ja milliseid andmestruktuure selleks kasutada. Selleks otstarbeks on võetud programmi sees kasutusele muutuja *g_Olukord* (1-baidine täisarv), mille väärtus vastab programmi faasile ja mida kontrollitakse *WM_PAINT*-teate töötlemisel. Muutuja g_Olukord väärtused *0* ja *1* vastavad tühjale peaaknale. Andmefaili avamisel omistatakse muutuja *g_Olukord* väärtuseks *2*. Sel juhul *WM_PAINT*-i saatmisel aknafunktsioonile kujundatakse peaaknasse tekst ning interferogrammi ja pimesignaali graafikud nii, nagu näha joonisel 4.2. Funktsiooni *BeginPaint()* väljakutsumisel saadetakse aknafunktsioonile veel teade *WM_ERASEBKGROUND*. Selle teate töötlemisel (lk lxxxiii) kontrollitakse muutuja *g_Olukord* väärtust ja vastavalt sellele värvitakse käsuga *FillRect()* üle peaakna taust.

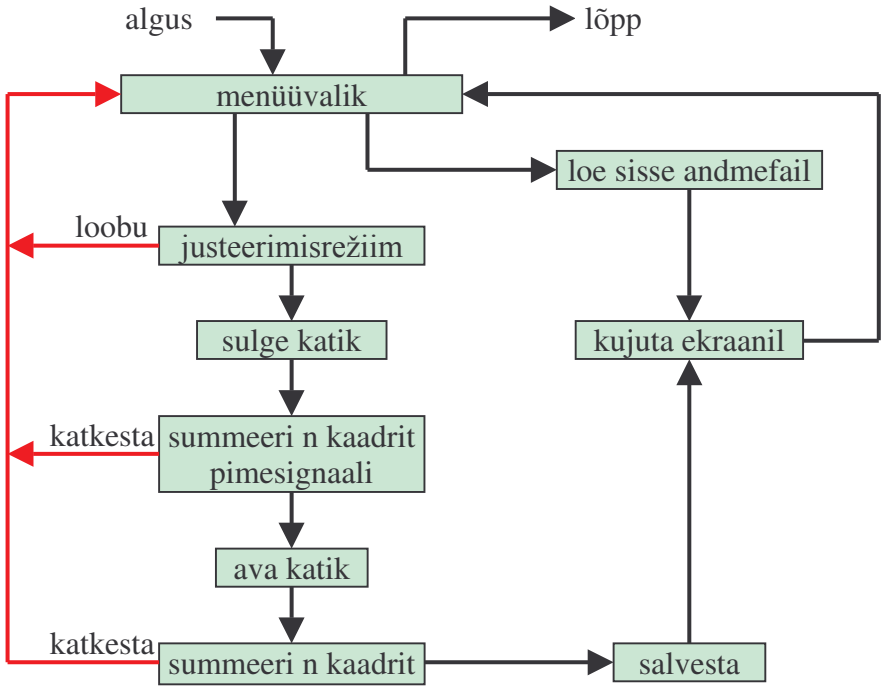
Kui kasutaja valib peaakna menüüst *Mõõda→Mõõda*, postitatakse aknafunktsioonile teade *WM_COMMAND* parameetriga *wParam*=201 (lk lxxxv) ja programm suundub nn justeerimisrežiimi. Funktsiooniga *capCreateCaptureWindowA()* luuakse peaaknasse uus aken videopildi kuvamiseks. Uue akna identifikaator on *g_hwCap*. Akna suurus ja paigutus kaamerapildi suhtes valitakse nii, et täpselt oleks näha monokromaatori pilu kujutises asuv



Joonis 4.3. Juhtprogrammi ekraanipilt justeerimisel

interferentsipilt. Videoparameetrite seadmine toimub süsteemis "Video for Windows" eeldefineeritud teadete (Messages) saatmisega videoaknale funktsiooni *SendMessage()* abil. Videoakna aknafunktsioon ja teadete töötlemise tsükkel organiseeritakse operatsioonisüsteemi poolt automaatselt. Videoaknale saadetavate teadete ja nende parameetrite (*lParam*, *wParam*, vt lisa IV) tähendused on defineeritud "VfW" päisefailides (vt ka lisa lk lxxviii).

Esimene videoaknale saadetav teade on *WM_CAP_DRIVER_CONNECT*, mille peale toimub TV-kaardi draiverite automaatne konfigureerimine. Kaamerapildi kuvamiseks videoaknas on süsteemis "VfW" kasutusel kaks režiimi: *Previev Mode* ja *Overlay Mode*. Esimesel juhul kantakse videokaader TV-kaardist esmalt süsteemi põhimälusse ja seejärel kujutatakse videoaknasse Windowsi GDI-funktsioonide abil. *Overlay*-skeemi puhul kantakse kaader TV-kaardist otse videokaardi puhvermälusse. Neid režiime saab valida, saates videoaknale vastavalt teate *WM_CAP_SET_PREVIEW* või *WM_CAP_SET_OVERLAY*. Justeerimisrežiimi puhul on kasutusel *Overlay Mode*, kuna ta võimaldab uuendada pilti videoaknas maksimaalse võimaliku kiirusega (kuni 30 kaadrit sekundis). Videopildi muude parameetrite, nagu kontrastsus, heledus, videoformaat jne, seadmiseks kasutatakse TV-kaardi draiveri funktsioone *Bt848Set...()* (lk



Joonis 4.4. Interferogrammi registreerimise protsess

lxxxv). Järgnevalt lisatakse põhiaknasse mitmesuguseid juhtelemente ja abitekste API-funktsiooni *CreateWindow()* abil (lk lxxxv). Juhtelementide ja videoakna paigutus justeerimisrežiimis on näha joonisel 4.3.

Nupu "*Alustan mõõtmist*" vajutamisel (joon. 4.3) postitatakse peaakna aknafunktsioonile teade *WM_COMMAND* parameetriga *wParam*=801 (lk lxxxvi). Videoaknale saadetakse funktsiooni *SendMessage()* abil teade *WM_CAP_SET_PREVIEW*, millega viiakse videosüsteem *Preview*-režiimi.

Videokaadri salvestamine ja töötlemine toimub järgmiselt. Teate *WM_CAP_SET_CALLBACK_FRAME* saatmisega (lk lxxxv) registreeriti "VfW" juurde funktsioon *Callback2()* (lk lxxxi), mis kutsutakse välja iga kord, kui TV-kaart on töödelnud järjekordse kaadri videosignaali. Funktsiooni sisenemisel antakse parameetrina kaasa viit muutujale tüübiga *PVIDEOHDR*, mille väljad sisaldavad informatsiooni videokaadri kohta. Muutuja väli *lpData* viitab mälupiirkonnale, kuhu on salvestatud massiivina kaadri pikslite heledusväärtused. Heleduse väärtused, mis vastavad horisontaalsele ribale rea numbrite *g_Rida1* ja *g_Rida2* vahel, kopeeritakse mälupuhvrisse, millele viitab muutuja *g_pBuf1*. Funktsiooni *CallBack2()* lõpus postitatakse peaakna aknafunktsioonile teade *WM_COMMAND*, *wParam*=802 ja tagastatakse juhtimine operatsioonisüsteemile. Selle teate saamisel (lk lxxxvii) summeeritakse kaadri pikslite heledusväärtused nii, et saadakse ühemõõtmeline numbrimassiiv, mis kajastab heleduse muutumist piki monokromaatori pilu kujutist (seega risti

interferentsimaksimumidega). Summeerimine toimub üle mitme järjestikuse kaadri, st iga kaadri puhul kutsutakse kõigepealt välja funktsioon *CallBack2()* (lk lxxxi) ja seejärel peaakna aknafunktsioon teatega *WM_COMMAND*, *wParam*=802. Liidetavate kaadrite arvu on eelnevalt määranud kasutaja (vt joonis 4.3). Videokaadri kujutamise eest videoaknas hoolitseb "VfW"-süsteem ise.

Operatsioonide loogiline järjekord interferentspildi mõõtmisel on toodud joonisel 4.4. Pärast menüüvalikut *Mõõda→Mõõda* kuvatakse joonisel 4.3 toodud ekraanipilt. Peaakna keskel on näha videoaken, milles kujutatakse (reaalajas) osa videokaamera pildist. Kasutaja justeerib pildi, muutes videokaamera ja lambi asendit ning monokromaatori pilu laiust, pöörates monokromaatori prismat ja muutes interferomeetri asendit reguleerkruvide abil. Ekraanil asuva *trackbar*-elemendi abil on võimalik muuta kaamerapildi heledust. Seejärel sisestab kasutaja vastavatesse lahtritesse mõõtmise parameetrid ja vajutab nuppu "*Alustan mõõtmist*". Programm kuvab modaalse dialoogi, millel on kiri "*Sulge katik!*" ja nupp *OK*. Sellele nupule vajutamise järel registreerib ja summeerib programm videopildi heledusväärtusi, kujutades üksikkaadrite ja summeeritud tulemust graafiliselt peaaknas. Kogunud vajaliku arvu kaadreid, kuvab programm dialoogiakna, milles palub monokromaatori katiku avada. Seejärel toimub kaadrite liitmine sarnaselt eelmisele, ainult heledussignaali väärtus salvestatakse teise muutujasse. Mõõtmise lõppedes salvestatakse tulemus kettafaili, mille formaat on toodud lisas III. Peaaken postitab iseendale teate *WM_COMMAND*, *wParam*=202. Sama teade postitatakse ka siis, kui kasutaja valib peaakna menüüst *Mõõda→Katkesta*. Selle peale saadab programm videoaknale teate *WM_CAP_DRIVER_DISCONNECT* (lk lxxxvi) ja hävitab videoakna käsuga *DestroyWindow()*. Seejärel siirdub programm andmefaili kuvamise režiimi (joonis 4.2).

Peaakna sulgemisel (valides peaakna menüüst *Fail→Välju* või süsteemimenüüst *Close* või klahvikombinatsiooni Alt+F4) saadetakse tema aknafunktsioonile teade *WM_CLOSE* (lk xc). Selle peale vabastab programm hõivatud operatsioonisüsteemi ressursid ja lõpetab iseenda teatetsükli käsuga *PostQuitMessage()*.

5 Kokkuvõte

Aastatel 1998..2004 on Tartu Ülikooli füüsikaosakonna spektroskoopia praktikumi seadmepark märgatavalt täienenud nii spektraalseadmete kui ka mõõtmiste automatiseerimise osas. Käesoleva magistritöö ülesandeks oli välja töötada ja realiseerida (nii programmide kui seadmete tasemel) automaatsed andmekogumissüsteemid töödele "Ramani hajumise uurimine", "Mõnede lihtsate aatomite spektrite struktuuri uurimine", "Kiirgusspektrite registreerimine fotoelektrilisel meetodil" ja "Spektrijoone ülipeenstruktuuri uurimine". Automaatse andmekogumissüsteemi loomise eesmärk on vähendada üliõpilase mehaanilise töö osa praktikumis ja võimaldada tal rohkem aega kulutada tulemuste mõtestamisele. Samas ei ole teadlikult viidud programmidesse katseandmete automaatset töötlemist, jättes selle osa üliõpilase lahendada. Käesolevas magistritöös sisaldub kokkuvõte nimetatud töö tulemustest. On esitatud töö käigus projekteeritud ja ehitatud seadmete skeemid ning kirjeldatud nende toimimise põhimõtteid. Peatükkides 2..4 on põhjalikult kirjeldatud mõõtekompleksid monokromaatoriga MDR-23 ja spektromeetriga DFS-52 ning seade spektrijoone ülipeenstruktuuri uurimiseks interferomeetri ja CCD-videokaamera baasil. Nende seadmete juhtimiseks on loodud originaalne tarkvara operatsioonisüsteemide MS Windows ja Linux graafilistele kasutajaliidestele. Programmide lähtetekstid on toodud lisades I..III. Punktis 2.6 on kirjeldatud eksperimentaalne kõvakettata terminalisüsteem mõõtreriistade juhtimiseks füüsika praktikumides.

Soovin avaldada sügavat tänu juhendajale Ilmar Rammole kannatlikkuse ja igakülgse abi eest käesoleva magistritöö valmimisel. Samuti soovin avaldada tänu Uno Veismannile, Silver Lättile ja Varpo Redile pikaajalise koostöö eest.

Ilmar Ansko

Tartu, 23.05.2005

6 Summary

Ilmar Ansko

Computer aided measurement systems for practical works in spectroscopy.

The technical modernization of practical works in spectroscopy by departement of physics at Tartu University is described. Practical works in spectroscopy were initiated in sixties and provided approximately 30 years without evolution of technical base. In 1998 the decision was made to modernize the equipment used and since 2004 all instrumentation is computer controlled. A new instrument control and data processing software based on the Windows and Linux graphical user interfaces was developed.

7 Kirjandus

1. Cerda, V., Guillermo, R. An Introduction to Laboratory Automation. John Wiley & Sons, Inc, 1990

2. Delphi Central - Article - Using the Serial Ports in Delphi 2.0 [http://www.delphi-central.com/serial.aspx] (23.05.2005)

3. Free Pascal - Home Page [http://www.freepascal.org] (23.05.2005)

4. GTK+ - The GIMP Toolkit [http://www.gtk.org] (23.05.2005)

5. Microchip Technology Inc., Home Page [http://www.microchip.com] (23.05.2005)

6. MSDN Library [http://msdn.microsoft.com/library/default.asp?url=/library/en-us/multimed/htm/_win32_video_for_windows.asp] (23.05.2005)

7. Rammo, Ilmar. Spektraalseadmed: optiline diapasoon. Tartu, 2002

8. theForger's Win32 API Tutorial [http://www.winprog.org/tutorial] (23.05.2005)

9. Tuvikene, Leon. Spektroskoopia praktikumi tööjuhendid. Tartu, 1969

10. Windows Documentation [http://www.piclist.com/techref/os/win/api/index.html] (23.05.2005)

11. Windows Techniques [http://www.relisoft.com/book/win/1hello.html] (23.05.2005)

12. Закамальдин, С. А., Викторов, Л. В. Прибор для измерения характеристик одноэлектронных фотоумножителей. Приборы и Техника Эксперимента, 5/1988, с. 141

13. Чернецкий, Ю. Н., Черный, С. А. Усилитель-ограничитель наносекундного диапазона. Приборы и Техника Эксперимента, 5/1986, с. 122

14. Шелевой, К. Д., Веников, А. А. Дифференциальный дискриминатор-счетчик импульсов. Приборы и Техника Эксперимента, 5/1984, с. 88

15. Федоров, А. Borland Pascal в среде Windows, 1993

16. Ansko, I. Arvutijuhitav kombinatsioonhajumisspektromeeter. Bakalaureusetöö. Tartu, 1999

Lisa I
Monokromaatoriga MDR-23 mõõtekompleks

PIC-kontrolleri programm

```
;*********************************************************
;**                                                     **
;**                       MDR.ASM                       **
;**                                                     **
;*********************************************************

; 22. m2rts 2005
; Kvarts 4 MHz, 9600 baudi

; RA0 - korgepinge clock
; RA1 - RS-232 RxD
; RA2 - RS-232 TxD
; RA3 - korgepinge data, nihkeregistri data
; RA4 - A/D data

; RB0 - korgepinge load
; RB1 - vaba
; RB2 - A/D clock
; RB3 - 1 nm andur
; RB4 - A/D rezhiim (hold/sample)
; RB5 - alguse/l6pu andur
; RB6 - nihkeregistri clock
; RB7 - 4094 load

        list            p=16F84         ;
        #include        <p16F84.inc>    ;

        __CONFIG    _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

;bitid:
KPCLOCK         equ     0               ;RA0
RxD             equ     1               ;RA1
TxD             equ     2               ;RA2
KPDATA          equ     3               ;RA3
ADDATA          equ     4               ;RA4
SHIFTDATA       equ     3               ;RA3
KPLOAD          equ     0               ;RB0
ADCLOCK         equ     2               ;RB2
NM              equ     3               ;RB3
ADCONTROL       equ     4               ;RB4
L6PP            equ     5               ;RB5
SHIFTCLOCK      equ     6               ;RB6
SHIFTLOAD       equ     7               ;RB7

;vabad registrid: 0x0C...0x2F
w_temp          equ     0x0C            ;konteksti savemiseks
status_temp     equ     0x0D            ;       "
;               equ     0x0E            ;
;               equ     0x0F            ;
high1           equ     0x10            ;m66tmistulemuse kogumiseks
low1            equ     0x11            ;       "
time            equ     0x12            ;faasilt faasile aeg
bytet           equ     0x13            ;
byter           equ     0x14            ;
RxCount         equ     0x15            ;
len             equ     0x16            ;
pointer         equ     0x17            ;
mootor          equ     0x18            ;
samme           equ     0x19            ;
adhigh          equ     0x20            ;
adlow           equ     0x21            ;
sek1            equ     0x22            ;
paus            equ     0x23            ;
sek2            equ     0x24            ;
shiftreg        equ     0x25            ;
poore           equ     0x26            ;
```

```
poordeid        equ     0x27            ;
;       ...                             ;
;               equ     0x2F            ;

;konstandid:
aeg1            equ     0xff
aeg2            equ     0xff
aeg3            equ     0x90

;*****************************************************************

        ORG             0x000                   ;koodi algus
        goto            Start                   ;

;*****************************************************************

        ORG             0x004                   ;interrupt
        movwf           w_temp                  ;
        movf            STATUS, w               ;
        movwf           status_temp             ;

        movf            status_temp, w          ;
        movwf           STATUS                  ;
        swapf           w_temp, f               ;
        swapf           w_temp, w               ;
        retfie

;*****************************************************************

Start
        clrwdt                                  ;
        bsf             STATUS, RP0             ;Bank1
        movlw           b'00010010'             ;
        movwf           TRISA                   ;
        movlw           b'00101010'             ;
        movwf           TRISB                   ;
        bcf             STATUS, RP0             ;Bank0

        bsf             PORTA, TxD              ;stoppbitt
        clrf            PORTB                   ;

        bcf             PORTA, KPCLOCK          ;
        clrf            mootor                  ;
        clrf            shiftreg                ;
        call            ToShiftreg              ;

;*****************************************************************
;
;               K2sud
;
;       1. n - nihkeregistrisse
;       2. v - mootor vasakule high:low sammu
;       3. p - mootor paremale        "
;       4. a - andurid
;       5. m - m66tmine
;       6. k - korgepinge (high, low)
;       7. f - faas maha
;       8. l - reeperi otsimine
;
;*****************************************************************

Commands
        call            RxByte                  ;

        movlw           'n'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            Nihkereg                ;

        movlw           'v'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            Vasakule                ;

        movlw           'p'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
```

```
        goto            Paremale                ;

        movlw           'a'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            Andurid                 ;

        movlw           'f'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            FaasMaha                ;

        movlw           'm'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            M66tmine                ;

        movlw           'k'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            K6rgepinge              ;

        movlw           'l'                     ;
        xorwf           byter, w                ;
        btfsc           STATUS, Z               ;
        goto            Reeper                  ;

        clrf            bytet                   ;vale k2sk
        call            TxByte                  ;

        goto            Commands                ;

;*****************************************************************

Valmis                                          ;
        movf            PORTB, w                ;
        movwf           bytet                   ;
        call            TxByte                  ;
        goto            Commands                ;uut k2sku ootama

;*****************************************************************

M66tmine                                        ;m66dab 16x
        clrf            high1                   ;
        clrf            low1                    ;
        movlw           d'16'                   ;
        movwf           RxCount                 ;tsykliloendur
M66tmine1                                       ;
        call            ADRead                  ;
        movf            adlow, w                ;
        addwf           low1, f                 ;noorem
        btfsc           STATUS, C               ;ylekanne?
        incf            high1, f                ;jah
        movf            adhigh, w               ;ei
        addwf           high1, f                ;vanem
        decfsz          RxCount, f              ;
        goto            M66tmine1               ;
        movf            low1, w                 ;saadab 2ra
        movwf           bytet                   ;
        call            TxByte                  ;
        movf            high1, w                ;
        movwf           bytet                   ;
        call            TxByte                  ;
        goto            Valmis                  ;

;*****************************************************************

ADRead                                          ;loeb AD
        clrf            adhigh                  ;
        bcf             PORTB, ADCONTROL        ;hoidmine
        bsf             PORTB, ADCLOCK          ;
        bcf             PORTB, ADCLOCK          ;
        bsf             PORTB, ADCLOCK          ;
        bcf             PORTB, ADCLOCK          ;
; d11
         bsf            PORTB, ADCLOCK          ;
         bcf            STATUS, C               ;
```

```
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adhigh, f             ;
; d10
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adhigh, f             ;
; d9
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adhigh, f             ;
; d8
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
       bsf              STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adhigh, f             ;
; d7
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d6
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d5
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d4
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d3
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d2
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d1
        bsf             PORTB, ADCLOCK        ;
        bcf             STATUS, C             ;
        btfsc           PORTA, ADDATA         ;
        bsf             STATUS, C             ;
        bcf             PORTB, ADCLOCK        ;
        rlf             adlow, f              ;
; d0
       bsf              PORTB, ADCLOCK        ;
       bcf              STATUS, C             ;
```

```
        btfsc           PORTA, ADDATA           ;
        bsf             STATUS, C               ;
        bcf             PORTB, ADCLOCK          ;
        rlf             adlow, f                ;

        bsf             PORTB, ADCLOCK          ;
        bcf             PORTB, ADCLOCK          ;
        bsf             PORTB, ADCLOCK          ;
        bcf             PORTB, ADCLOCK          ;
        bsf             PORTB, ADCONTROL        ;
        return                                  ;

;*****************************************************************

Andurid                                         ;
        goto             Valmis                 ;

;*****************************************************************

Nihkereg                                        ;
        movlw           0x0F                    ;
        andwf           shiftreg, f             ;
        call            RxByte                  ;
        iorwf           shiftreg, f             ;
        call            ToShiftreg              ;
        goto            Valmis                  ;

;*****************************************************************

K6rgepinge                                      ;
        call            RxByte                  ;
        movwf           adlow                   ;
        call            RxByte                  ;
        movwf           adhigh                  ;
        movlw           0x08                    ;
        movwf           RxCount                 ;
K6rgepinge1                                     ;
        bsf             PORTA, KPDATA           ;
        btfss           adlow, 0                ;
        bcf             PORTA, KPDATA           ;
        bsf             PORTA, KPCLOCK          ;
        nop                                     ;
        bcf             PORTA, KPCLOCK          ;
        rrf             adlow, f                ;
        decfsz          RxCount, f              ;
        goto            K6rgepinge1             ;
        movlw           0x04                    ;
        movwf           RxCount                 ;
K6rgepinge2                                     ;
        bsf             PORTA, KPDATA           ;
        btfss           adhigh, 0               ;
        bcf             PORTA, KPDATA           ;
        bsf             PORTA, KPCLOCK          ;
        nop                                     ;
        bcf             PORTA, KPCLOCK          ;
        rrf             adhigh, f               ;
        decfsz          RxCount, f              ;
        goto            K6rgepinge2             ;
        nop                                     ;
        bsf             PORTB, KPLOAD           ;
        nop                                     ;
        nop                                     ;
        bcf             PORTB, KPLOAD           ;
        goto            Valmis                  ;

;*****************************************************************

Reeper                                          ;
        movlw           d'255'                  ;aeg kahe sammu vahel
        movwf           time                    ;
Reeper1                                         ;
        decf            mootor, f               ;samm paremale
        movlw           0xF0                    ;
        andwf           shiftreg, f             ;
        movlw           0x03                    ;
        andwf           mootor, f               ;
        call            Faas                    ;
```

```
        iorwf           shiftreg, f             ;
        call            ToShiftreg              ;
        call            Passi                   ;
        btfss           PORTB, L6PP             ;kontrollib l6puandurit
        goto            Valmis                  ;katkestab
        btfss           PORTB, NM               ;
         goto           Reeper1                 ;
        call            Passi                   ;
        call            Passi                   ;
        call            Passi                   ;
Reeper2                                         ;
        incf            mootor, f               ;samm vasakule
        movlw           0xF0                    ;
        andwf           shiftreg, f             ;
        movlw           0x03                    ;
        andwf           mootor, f               ;
        call            Faas                    ;
        iorwf           shiftreg, f             ;
        call            ToShiftreg              ;
        call            Passi                   ;
        btfss           PORTB, L6PP             ;kontrollib l6puandurit
        goto            Valmis                  ;katkestab
        btfsc           PORTB, NM               ;
         goto           Reeper2                 ;

        goto            Valmis                  ;

;*********************************************************************

Vasakule                                        ;mootor vasakule
        call            RxByte                  ;noorem bait
        movwf           adlow                   ;
        call            RxByte                  ;vanem bait
        movwf           adhigh                  ;
        movlw           d'255'                  ;aeg kahe sammu vahel
        movwf           time                    ;
Vasakule1                                       ;
        movf            adlow, f                ;
        btfss           STATUS, Z               ;kas noorem=0?
        goto            Vasakule2               ;ei
        movf            adhigh, f               ;jah
        btfsc           STATUS, Z               ;kas vanem=0?
        goto            Valmis                  ;m6lemad on nullid
        decf            adhigh, f               ;
Vasakule2                                       ;noorem pole veel 0
        decf            adlow, f                ;
        incf            mootor, f               ;
        movlw           0xF0                    ;
        andwf           shiftreg, f             ;
        movlw           0x03                    ;
        andwf           mootor, f               ;
        call            Faas                    ;tagastab faasid
        iorwf           shiftreg, f             ;kirjutab nihkeregistrisse
        call            ToShiftreg              ;
        call            DecTime                 ;
        call            Passi                   ;passib natuke
        btfss           PORTB, L6PP             ;kontrollib l6puandurit
        goto            Valmis                  ;katkestab kerimise
         goto           Vasakule1               ;

;*********************************************************************

Paremale                                        ;mootor paremale
        call            RxByte                  ;noorem bait
        movwf           adlow                   ;
        call            RxByte                  ;vanem bait
        movwf           adhigh                  ;
        movlw           d'255'                  ;aeg kahe sammu vahel
        movwf           time                    ;
Paremale1                                       ;
        movf            adlow, f                ;
        btfss           STATUS, Z               ;kas noorem=0?
        goto            Paremale2               ;ei
        movf            adhigh, f               ;jah
        btfsc           STATUS, Z               ;kas vanem=0?
        goto            Valmis                  ;m6lemad on nullid
        decf            adhigh, f               ;
```

```
Paremale2                                   ;noorem pole veel 0
       decf          adlow, f               ;
       decf          mootor, f              ;
       movlw         0xF0                   ;
       andwf         shiftreg, f            ;
       movlw         0x03                   ;
       andwf         mootor, f              ;
       call          Faas                   ;tagastab faasid
       iorwf         shiftreg, f            ;kirjutab nihkeregistrisse
       call          ToShiftreg             ;
       call          DecTime                ;
       call          Passi                  ;passib natuke
       btfss         PORTB, L6PP            ;kontrollib l6puandurit
       goto          Valmis                 ;katkestab kerimise
        goto         Paremale1              ;

;*********************************************************************

ToShiftreg                                  ;
       movlw         0x08                   ;
       movwf         RxCount                ;
       movf          shiftreg, w            ;
       movwf         byter                  ;
ToShiftreg1                                 ;
       bsf           PORTA, SHIFTDATA       ;
       btfss         byter, 7               ;
       bcf           PORTA, SHIFTDATA       ;
       bsf           PORTB, SHIFTCLOCK      ;
       bcf           PORTB, SHIFTCLOCK      ;
       rlf           byter, f               ;
       decfsz        RxCount, f             ;
       goto          ToShiftreg1            ;

       bsf           PORTB, SHIFTLOAD       ;
       bcf           PORTB, SHIFTLOAD       ;
       return                               ;valmis

;*********************************************************************

FaasMaha                                    ;
       movlw         0xF0                   ;
       andwf         shiftreg, f            ;
       call          ToShiftreg             ;
       call          Passi                  ;
       goto          Valmis                 ;

;*********************************************************************

DecTime                                     ;
       movlw         0x40                   ;
       subwf         time, w                ;
       btfss         STATUS, Z              ;
       decf          time, f                ;
       return                               ;

;*********************************************************************

IncTime                                     ;?????
       movlw         0xFF                   ;
       subwf         time, w                ;
       btfss         STATUS, Z              ;
       incf          time, f                ;
       return                               ;

;*********************************************************************

Passi                                       ;
       movf          time, w                ;
       movwf         paus                   ;
Passi1                                      ;
       nop                                  ;
       nop                                  ;
       nop                                  ;
       nop                                  ;
       nop                                  ;
       nop                                  ;
       nop                                  ;
```

```
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        decfsz          paus, f         ;
        goto            Passi1          ;
        return                          ;

;*****************************************************************

Delay                                   ;W*3+3=(W+1)*3us
        movwf           paus            ;4
Delay1                                  ;
        decfsz          paus,f          ;5      8       11
        goto            Delay1          ;6      9       12
        return                          ;94

;*****************************************************************

RxByte                                  ;104us
        btfsc           PORTA, RxD      ;ootame startbitti
        goto            RxByte          ;
        movlw           d'17'           ;
        call            Delay           ;
        btfsc           PORTA, RxD      ;startbitt on alles?
        goto            RxByte          ;ei
        nop                             ;jah
        nop                             ;4
        nop                             ;5
        movlw           8               ;6
        movwf           RxCount         ;7
RxByte2                                 ;
        movlw           d'30'           ;8      112     214
        call            Delay           ;9      113     215
        bcf             STATUS, C       ;103    207     309
        btfsc           PORTA, RxD      ;104    208     310
        bsf             STATUS, C       ;105    207     311
        rrf             byter, f        ;106    208     312
        nop                             ;107    209     ...
        nop                             ;108    210
        decfsz          RxCount, f      ;109    211
        goto            RxByte2         ;110    212

        movlw           d'30'           ;
        call            Delay           ;stopp-bitt
        btfss           PORTA, RxD      ;ootame tyhja liini
        goto            $-1             ;

        movf            byter, w        ;
        return                          ;

;*****************************************************************

TxByte                                  ;
        bcf             PORTA, TxD      ;startbitt
        movlw           d'30'           ;2
        call            Delay           ;96
        movlw           8               ;97
        movwf           RxCount         ;98
        nop                             ;99
TxByte1                                 ;
        btfss           bytet, 0        ;100    204
        goto            TxByte2         ;101    205
        nop                             ;102    206
        nop                             ;103    207
        bsf             PORTA, TxD      ;104    208
        goto            TxByte3         ;105    ...
TxByte2                                 ;
        nop                             ;103
        bcf             PORTA, TxD      ;104
        nop                             ;105
        nop                             ;106
TxByte3                                 ;
```

```
        rrf             bytet, f                ;107
        movlw           d'29'                   ;108
        call            Delay                   ;109
        nop                                     ;200
        decfsz          RxCount, f              ;201
        goto            TxByte1                 ;202

        bsf             PORTA, TxD              ;stoppbitt
        movlw           d'32'                   ;
        call            Delay                   ;
        rrf             bytet, f                ;keerab 6igeks
        return                                  ;

;*****************************************************************

        ORG             0x300                   ;
Faas                                            ;faasipinged mootorile
        movlw           HIGH Faas               ;
        movwf           PCLATH                  ;
        movf            mootor, w               ;
        addwf           PCL, f                  ;
        retlw           b'00001001'             ;
        retlw           b'00000011'             ;
        retlw           b'00000110'             ;
        retlw           b'00001100'             ;

;*****************************************************************

        END                                     ;
```

Windows-keskonna juhtprogramm

```
program MDR_95;

{fp:}
(*{$APPTYPE GUI}
uses Windows, SysUtils;*)
{Delphi:}
uses Windows, SysUtils, CommDlg, CommCtrl;

type massiiv = array[0..2000] of longint;
     pmassiiv = ^massiiv;
     v2rv = array[0..767] of longint;
     pv2rv = ^v2rv;
     andmed = record
              algus, l6pp, kuup, takisti: LONGINT;
              samm, kordus, pinge, punkte, max, xnr, x0, dx3, dx4, nih: longint;
              k1max, k1min: real;
              k1, dx1, dx2, dx5: real;
              komm: array[0..255] of char;
              aeg: array[0..20] of char;
              nimi, laiend: array[0..50] of char;
              m: pmassiiv;
              v: pv2rv;
              end;
     pandmed = ^andmed;

var g_hwMain, g_hwKoordinaat, g_hwT66, g_hwAva: HWND;
    g_hwUpDownAva, g_hwUpDownT66: HWND;
    g_DataPath: array[0..255] of char;
    g_x, g_l1: longint;
    g_Punkt: integer;
    g_hm1: hmenu;
    {g_DCPr: hDC;}
    g_Font1: HFONT;
    g_M66dan: LONGBOOL;
    g_m1, g_m2: massiiv;
    g_AvaHeader, g_DefaultHeader, g_T66Header: andmed;
    {TPD1 : TPrintDlg;
    DI1 : TDocInfo;
    hp1 : hpen;
    HR1 : hRgn;}
    g_TRGraafik, g_TRxtelg: TRECT;
    g_HRGraafik: hRGN;
    g_hCOM: tHandle;
    g_dcb: tDCB;
    g_CTO: tCommTimeOuts;
    dw1, g_Viivitus: dword;
    g_htr1: tHandle;
    g_HallPintsel: HBRUSH;
    g_HallSulg: HPEN;
    g_Sodi: record yks: LONGBOOL; nr: longint; end;
    g_AvaV2rv, g_T66V2rv: v2rv;

const g_Katkestan: boolean = FALSE;
      g_M66tmiseTyyp: byte = 0;

{$R M95.res}

procedure piip; begin messagebeep($ffff); end;

function PordiViivitus(dw : dword) : boolean;
begin
PordiViivitus:=FALSE;
with g_CTO do begin
              ReadIntervalTimeout:=0;
              ReadTotalTimeoutMultiplier:=dw;
              ReadTotalTimeoutConstant:=0;
              WriteTotalTimeoutMultiplier:=0;
              WriteTotalTimeoutConstant:=0;
              end;
if not SetCommTimeouts(g_hCOM,g_CTO) then exit;
PordiViivitus:=TRUE;
end;
```

```
function AvaPort: LONGBOOL;
begin
AvaPort:=FALSE;
g_hCOM:=CreateFile('COM1',GENERIC_READ+GENERIC_WRITE,0,nil,
                   OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,0);
if g_hCOM=INVALID_HANDLE_VALUE then exit;
if not SetupComm(g_hCOM,16,16) then exit;
if not GetCommState(g_hCOM,g_dcb) then exit;
with g_dcb do begin BaudRate:=9600; ByteSize:=8; Parity:=0; StopBits:=0; end;  {with}
if not SetCommState(g_hCOM,g_dcb) then exit;
g_Viivitus:=1000;
if not PordiViivitus(g_Viivitus) then exit;
AvaPort:=TRUE;
end;

function ResetPort: LONGBOOL;
begin
ResetPort:=FALSE;
EscapeCommFunction(g_hCOM,SETDTR);  {reset PIC-le}
sleep(100);
EscapeCommFunction(g_hCOM,CLRDTR);
sleep(100);
if not PurgeComm(g_hCOM,PURGE_TXABORT+PURGE_RXABORT+PURGE_TXCLEAR+PURGE_RXCLEAR) then exit;
ResetPort:=TRUE;
end;

function SulePort: LONGBOOL;
begin
if CloseHandle(g_hCOM) then SulePort:=TRUE else SulePort:=FALSE;
end;

function KoordinaatProc(DlgWin: hWnd; Msg, wParam, lParam: longint): longint; stdcall;
var i1: integer;
    bl: longbool;
    s1: string;
begin
KoordinaatProc:=1;
case Msg
of WM_COMMAND: begin
               case wParam
               of 102: begin  {OK}
                       i1:=0;
                       i1:=GetDlgItemInt(DlgWin,101,bl,FALSE);
                       if ((i1>200) and (i1<800))
                       then begin
                            EndDialog(DlgWin,i1);
                            exit;
                            end
                       else begin
                            i1:=MessageBox(DlgWin,'Peab olema 200..800 nm',
                                           'Viga:',MB_OK+MB_ICONEXCLAMATION);
                            SendDlgItemMessage(DlgWin,101,EM_SETSEL,0,-1);
                            SetFocus(GetDlgItem(DlgWin,101));
                            exit;
                            end;
                       end;
                  103: begin  {Cancel}
                       EndDialog(DlgWin,0);
                       exit;
                       end;
                  104: begin  {uuesti}
                       EndDialog(DlgWin,100);
                       exit;
                       end;
                  666: begin  {algusprotseduurid}
                       SendDlgItemMessage(DlgWin,101,EM_SETSEL,0,-1);
                       SetFocus(GetDlgItem(DlgWin,101));
                       exit;
                       end;
                  else begin KoordinaatProc:=0; exit; end;
                  end;  {case}
               end;  {WM_COMMAND}
   WM_INITDIALOG: begin  {160x70}
                  g_hwKoordinaat:=DlgWin;
                  str((g_x div 300)-1,s1);
                  if ((g_x<300*300) or (g_x>300*800)) then s1:='';  {!!!!!!}
```

```
                  s1:=s1+#0;
                  SetDlgItemText(DlgWin,101,@s1[1]);
                  SetWindowPos(DlgWin,0,(GetSystemMetrics(SM_CXFULLSCREEN)-320) shr 1,
                               (GetSystemMetrics(SM_CYFULLSCREEN)-140) shr 1,
                               0,0,SWP_NOSIZE+SWP_NOZORDER);
                  PostMessage(DlgWin,WM_COMMAND,666,0);
                  exit;
                  end;
   WM_CLOSE: begin
             KoordinaatProc:=0;
             EndDialog(DlgWin,0);
             exit;
             end
   else begin KoordinaatProc:=0; exit; end;
end;  {case}
end;  {KoordinaatProc}

function T66Proc0(pl: plongint): DWORD; stdcall;  {Init}
var b: byte;
    hw: hWnd;
    dw: dword;
    bl: boolean;
    buf_out: array[1..10] of char;
label 1, 2, 3;
begin
T66Proc0:=0;
hw:=pl^;
buf_out[1]:='n';  {takisti}
buf_out[2]:=char($20);
bl:=WriteFile(g_hCOM,buf_out,2,dw,nil); if ((not bl) or (dw<>2)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
buf_out[1]:='k';  {k6rgepinge}
dw:=round(1.53*g_T66Header.pinge);
Move(dw,buf_out[2],2);
bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,501,0);  {k6ik korras}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {PIC ei vasta}
ExitThread(0);
exit;
2:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,2);  {keris piirest v@lja}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc0}

function T66Proc1(pl: plongint): DWORD; stdcall;  {m66tmine ja sammumine}
var j, punkt: integer;
    b: byte;
    hw: hWnd;
    dw, dw1: dword;
    l: longint;
    bl: boolean;
    buf_out: array[1..10] of char;
label 1, 2, 3;
begin
T66Proc1:=0;
hw:=pl^;
g_Viivitus:=1000+round(1.6*g_T66Header.samm);
if not PordiViivitus(g_Viivitus) then goto 1;
{-------------------------------------------------------------------------}
g_Punkt:=0;
punkt:=0;
l:=0;
buf_out[1]:='m';
buf_out[2]:='v';  {n sammu vasakule}
Move(g_T66Header.samm,buf_out[3],2);
for j:=1 to g_T66Header.kordus
```

```
do begin  {esimene punkt}
   if g_Katkestan then goto 3;
   bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
   bl:=ReadFile(g_hCOM,dw1,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;  {lo,hi,OK}
   l:=l+(dw1 and $0000ffff);
   end;
g_m1[0]:=l;
if l<>0 then g_T66Header.k1:=180/l else g_T66Header.k1:=1;  {?????}
g_T66Header.max:=0;
{PostMessage(hw,WM_COMMAND,502,0);}
repeat
if g_Katkestan then goto 3;
bl:=WriteFile(g_hCOM,buf_out[2],3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if (b and 32)=0 then goto 2;  {keris piirest v@lja}
inc(g_x,g_T66Header.samm);
inc(punkt);
if ((punkt>560) and (g_M66tmiseTyyp=2)) then punkt:=0;
l:=0;
for j:=1 to g_T66Header.kordus
do begin  {m66tmine ja keskmistamine}
   if g_Katkestan then goto 3;
   bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
   bl:=ReadFile(g_hCOM,dw1,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;  {lo,hi,OK}
   l:=l+(dw1 and $0000ffff);
   end;
g_m1[punkt]:=l;
PostMessage(hw,WM_COMMAND,502,0);  {yks punkt m66detud}
if g_Katkestan then goto 3;
until punkt=g_T66Header.punkte-1;
{-------------------------------------------------------------------------------}
g_Viivitus:=1000;
if not PordiViivitus(g_Viivitus) then goto 1;  {?????}
PostMessage(hw,WM_COMMAND,503,0);  {valmis}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {pordi jama}  {!!!!!}
ExitThread(0);
exit;
2:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,2);  {keris piirest v@lja}  {!!!!!}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc1}

function T66Proc3(pl: plongint): DWORD; stdcall;  {l6petamine}
var b: byte;
    hw: hWnd;
    dw, dw1: dword;
    bl: boolean;
    buf_out: array[1..10] of char;
label 1, 3;
begin
T66Proc3:=0;
hw:=pl^;
buf_out[1]:='k';  {pinge maha}
buf_out[2]:=#0;
buf_out[3]:=#0;
bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,dw1,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
buf_out[1]:='f';  {faas maha}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,504,0);  {k6ik korras}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {pordi jama}  {!!!!!}
ExitThread(0);
```

```
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc3}

function T66Proc4(pl: plongint): DWORD; stdcall;  {liikumine koordinaadile}
var i: integer;
    b: byte;
    l: longint;
    hw: hWnd;
    dw: dword;
    bl: boolean;
    buf_out: array[1..10] of char;
label 1, 2, 3, 4;
begin
T66Proc4:=0;
hw:=pl^;
(*buf_out[1]:='f';  {faas maha}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if (b and 32)=0 then goto 2;  {keris piirest v@lja}*)
4:
buf_out[1]:='l';  {reeperi otsimine}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if (b and 32)=0 then goto 2;  {keris piirest v@lja}
{}
l:=DialogBox(hInstance,'KOORDINAAT',hw,@KoordinaatProc);
if l=0 then goto 3;
if l=100 then goto 4;
g_x:=300*l{+60};
l:=g_T66Header.algus-g_x;
g_Viivitus:=1000+round(1.6*abs(l));
if not PordiViivitus(g_Viivitus) then goto 1;
{-----------------------}  {liikumine koordinaadile}
if l>0 then buf_out[1]:='v' else buf_out[1]:='p';
buf_out[2]:=#255;
buf_out[3]:=#255;
for i:=1 to (abs(l) div 65535)
do begin
   if g_Katkestan then goto 3;
   bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
   bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
   if (b and 32)=0 then goto 2;  {keris piirest v@lja}
   if (l<0) then g_x:=g_x-65535 else g_x:=g_x+65535;
   sleep(1000);
   end;
if g_Katkestan then goto 3;
if l>0 then buf_out[1]:='v' else buf_out[1]:='p';
l:=abs(g_T66Header.algus-g_x);
buf_out[2]:=char(l mod 256);
buf_out[3]:=char(l div 256);
bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if (b and 32)=0 then goto 2;  {keris piirest v@lja}
g_x:=g_T66Header.algus;
{-----------------------}
g_Viivitus:=1000;
if not PordiViivitus(g_Viivitus) then goto 1;
if g_Katkestan then goto 3;
if g_M66tmiseTyyp=2
then begin  {faas maha}
     buf_out[1]:='f';
     bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
     bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
     if (b and 32)=0 then goto 2;  {keris piirest v@lja}
     end;
PostMessage(hw,WM_COMMAND,506,0);  {k6ik korras}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {pordi jama}  {!!!!!}
ExitThread(0);
exit;
2:
```

```
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,2);  {keris piirest v@lja}  {!!!!!}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc4}

function OnNimi(p: pChar): LONGBOOL;  {!!!!!}
var i, j: integer;
    buf: array[0..50] of char;
begin
StrLCopy(buf,p,50);
OnNimi:=FALSE;
i:=StrLen(buf);
if i=0 then exit;
for j:=1 to i-1 do if (not(buf[j] in ['A'..'Z','a'..'z','0'..'9','_','-',' '])) then exit;
OnNimi:=TRUE;
end;

function OpenFileName(b: byte; var nimi: string): LONGBOOL;
var f: TOpenFileName;
    buf1, buf2: array[0..255] of char;
begin
OpenFileName:=FALSE;
FillChar(f,sizeOf(f),#0);
FillChar(buf1,SizeOf(buf1),#0);
FillChar(buf2,SizeOf(buf2),#0);
buf1:='dat file'+#0+'*.dat'+#0+'Any File'+#0+'*.*'+#0+#0;
f.lStructSize:=sizeof(f);
f.lpstrFilter:=@buf1;
f.nMaxFile:=sizeof(buf2);
f.lpstrFile:=@buf2;
f.lpstrInitialDir:=@g_DataPath;
{fp:}
{if b=1 then OpenFileName:=GetOpenFileName(@f) else OpenFileName:=GetSaveFileName(@f);}
{Delphi:}
if b=1 then OpenFileName:=GetOpenFileName(f) else OpenFileName:=GetSaveFileName(f);
nimi:=StrPas(@buf2)+#0;
end;

function L6iguReal(r : real; b : byte) : string;
var l : longint;
    s : string;
begin
l:=trunc(r);
if l=r then str(l,s) else str(r:1:b,s);
L6iguReal:=s+#0;
end;

function StrToLong(var s : string; var l : longint) : LongBool;  {leiab stringist t@isarvu}
var i, j : integer;
begin
StrToLong:=FALSE;
i:=1;
while (not (s[i] in ['0'..'9'])) do begin s[i]:=#32; inc(i); end;
j:=i;
while (s[i] in ['0'..'9']) do inc(i);
s:=copy(s,j,i-j);
val(s,l,i);
if i=0 then StrToLong:=TRUE;
end;

function TextToFloat(p: pChar; l: integer; var r: real): boolean;
var i, j, k : integer;
    c : char;
    r1, r2 : extended;
    buf1, buf2, buf3 : array[0..20] of char;
label 1, 2, 3;
begin
TextToFloat:=FALSE;
Move(p^,buf1,20);
i:=0;
j:=0;
k:=0;
1:
```

```
c:=buf1[i];
if i>10 then exit;  {jama}
if (c in ['0'..'9']) then begin buf2[j]:=c; inc(j); inc(i); goto 1; end;
if c='.' then begin buf2[j]:=#0; inc(i); goto 2; end;
if c=#0 then begin buf2[j]:=#0; buf3[0]:=#0; goto 3; end;
exit;  {jama}
2:
c:=buf1[i];
if i>10 then exit;  {jama}
if (c in ['0'..'9']) then begin buf3[k]:=c; inc(k); inc(i); goto 2; end;
if c=#0 then begin buf3[k]:=#0; goto 3; end;
exit;  {jama}
3:
r1:=StrToIntDef(StrPas(@buf2),0);
r2:=StrToIntDef(StrPas(@buf3),0);
for i:=1 to k do r2:=r2/10;
r:=r1+r2;
TextToFloat:=TRUE;
end;

function Salvesta(nimi : pChar) : boolean;
var f : text;
    i : integer;
    r : real;
begin
Salvesta:=FALSE;
Assign(f,StrPas(nimi));
{$I-} ReWrite(f);  {$I+}
if IOResult<>0 then exit;
with g_T66Header
do begin
   writeln(f,'Win_MDR andmefail');
   writeln(f,aeg);
   writeln(f,'Algus: ',algus div 300,' nm');
   writeln(f,'Samm: ',samm,' /300 nm');
   writeln(f,'Katsepunkte: ',punkte);
   writeln(f,'Kordsus: ',kordus);
   writeln(f,'Anoodpinge: ',pinge,' V');
   writeln(f,'Koormustakisti: ',takisti);
   writeln(f,'Kommentaar: ',komm);
   for i:=1 to 7 do writeln(f);
   for i:=0 to punkte-1 do begin
                           r:=(algus div 300)+i*samm/300;
                           writeln(f,r:2:2,' ',g_m1[i]);
                           end;
   end;  {with}
Close(f);
end;  {Salvesta}

function LoeSisse(nimi: pChar; var viga: integer): LONGBOOL;
var f: text;
    i: integer;
    r: real;
    s: string;
    bl: LongBool;
label 1;
begin
LoeSisse:=FALSE;
Assign(f,nimi);
s:=ExtractFileName(StrPas(nimi));  {nimi koos laiendiga}
StrPCopy(g_AvaHeader.nimi,s);
IOResult;
{$I-} Reset(f);
if IOResult<>0 then begin viga:=-1; goto 1; end;
with g_AvaHeader
do begin
   viga:=0;
   readln(f,s);  {ID}
   if s<>'Win_MDR andmefail' then goto 1;
   inc(viga);
   readln(f,aeg);
   inc(viga);
   readln(f,s);
   bl:=StrToLong(s,algus);
   if ((algus<200) or (algus>800) or (not bl)) then goto 1;
   algus:=300*algus;
   inc(viga);
```

```
   readln(f,s);
   bl:=StrToLong(s,samm);
   if ((samm<3) or (samm>9999) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);
   bl:=StrToLong(s,punkte);
   if ((punkte<3) or (punkte>9999) or (not bl)) then goto 1;
   l6pp:=algus+(punkte-1)*samm;
   inc(viga);
   readln(f,s);
   bl:=StrToLong(s,kordus);
   if ((kordus<10) or (kordus>9999) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);
   bl:=StrToLong(s,pinge);
   if ((pinge<0) or (pinge>2000) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);
   bl:=StrToLong(s,takisti);
   if ((takisti<0) or (takisti>3) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);  {13}
   s:=copy(s,13,255);
   StrPCopy(komm,s);
   for i:=1 to 7 do readln(f);
   max:=0;
   for i:=0 to punkte-1 do begin
                           inc(viga);
                           readln(f,r,g_m2[i]);
                           if g_m2[i]>g_m2[max] then max:=i;
                           end;
   end;  {with}
Close(f); {$I+}
if IOResult<>0 then begin viga:=-2; goto 1; end;
LoeSisse:=TRUE;
exit;
1: Close(f); {$I+}
end;  {LoeSisse}

procedure ArvutaV2rvid(a: pandmed);
var buf: array[0..3] of byte;
    r1, r2, lambda: real;
    i1: integer;
begin
for i1:=0 to 550
do begin
   r2:=(a^.l6pp-a^.algus)/530;
   lambda:=(a^.algus+i1*r2)/300;  {nm}
   buf[3]:=0;
   r1:=255*exp(-sqr(lambda-600)/10000);
   buf[0]:=lo(loword(round(r1)));
   r1:=255*exp(-sqr(lambda-550)/10000);
   buf[1]:=lo(loword(round(r1)));
   r1:=255*exp(-sqr(lambda-450)/5000);
   buf[2]:=lo(loword(round(r1)));
   Move(buf,a^.v^[i1],4);
   end;
end;  {ArvutaV2rvid}

procedure ArvutaParameetrid(a: pandmed);
var r1: real;
begin
with a^
do begin
   punkte:=((l6pp-algus) div samm)+1;
   l6pp:=algus+samm*(punkte-1);
   if m^[max]<>0 then k1:=180/m^[max] else k1:=1;
   k1max:=k1*1000;
   k1min:=k1/5;
   dx1:=530/(punkte-1);  {x-nihe pikslites}
   r1:=samm*(punkte-1)/300;
   dx5:=530/r1;  {pikslit nanomeetrile}
   r1:=ln(r1)/ln(10);  {log(l6pp-algus)}
   if r1>=0 then r1:=trunc(r1) else r1:=trunc(r1)-1;  {ceil ei t66ta}
   dx2:=round(exp(ln(10)*r1));  {jaotise suurus}
   xnr:=trunc((l6pp-algus)/(300*dx2));  {jaotiste arv}
   if xnr<2 then dx2:=dx2/5 else if xnr<5 then dx2:=dx2/2;
```

```
   r1:=samm*(punkte-1)/(300*dx2);
   xnr:=round(r1);  {jaotiste arv}
   dx5:=dx5*dx2;
   x0:=round(algus/300);
   dx3:=1;  {?????}
   end;  {with}
end;  {ArvutaParameetrid}

procedure T2idaLahtrid(hw: HWND; a: pandmed);
var r1: real;
    s1: string;
begin
with a^
do begin
   SetDlgItemText(hw,105,@nimi);
   SetDlgItemText(hw,106,@aeg);
   r1:=algus div 300; s1:=L6iguReal(r1,1);
   SetDlgItemText(hw,107,@s1[1]);
   r1:=l6pp/300; s1:=L6iguReal(r1,1);
   SetDlgItemText(hw,108,@s1[1]);
   SetDlgItemText(hw,109,@komm);
   str(punkte,s1); s1:=s1+#0; SetDlgItemText(hw,110,@s1[1]);
   str(kordus,s1); s1:=s1+#0; SetDlgItemText(hw,111,@s1[1]);
   str(pinge,s1); s1:=s1+#0; SetDlgItemText(hw,116,@s1[1]);
   CheckDlgButton(hw,115,0);
   s1:='1M'+#0;
   SendDlgItemMessage(hw,117,CB_ADDSTRING,0,longint(@s1[1]));
   s1:='3M'+#0;
   SendDlgItemMessage(hw,117,CB_ADDSTRING,0,longint(@s1[1]));
   s1:='10M'+#0;
   SendDlgItemMessage(hw,117,CB_ADDSTRING,0,longint(@s1[1]));
   SendDlgItemMessage(hw,117,CB_SETCURSEL,takisti-1,0);
   r1:=samm/300; s1:=L6iguReal(r1,1);
   SetDlgItemText(hw,113,@s1[1]);
   end;
end;  {T2idaLahtrid}

function Prindi(hw: HWND; a: pandmed): LONGBOOL;
begin
{***}
(*{PrintDlg(TPD1);}
g_DCPr:=CreateDC('EPSON9','Epson FX-80 (Robotron)','LPT1',nil);
if g_DCPr=0 then goto 2;
DI1.cbSize:=100;
DI1.lpszDocName:='Graafiku printimine'+#0+#0;
DI1.lpszOutput:=nil;
if StartDoc(g_DCPr,DI1)<=0 then goto 2;
if StartPage(g_DCPr)<=0 then goto 2;
{*****}
str(trunc(g_Header3.algus/48),s1);
s1:='Spektri algus: '+s1+' cm^-1';
TextOut(g_DCPr,50,120,@s1[1],length(s1));
                        str(round(g_Header3.l6pp/48),s1);
                        s1:='Spektri l6pp: '+s1+' cm^-1';
                        TextOut(g_DCPr,50,140,@s1[1],length(s1));
                        str(g_Header3.samm,s1);
                        s1:='Skaneerimise samm: '+s1+'/48 cm^-1';
                        TextOut(g_DCPr,50,160,@s1[1],length(s1));
                        str(g_Header3.punkte,s1);
                        s1:='Katsepunktide arv: '+s1;
                        TextOut(g_DCPr,480,120,@s1[1],length(s1));
                        {str(round(g_Header3.aeg*1000/70),s1);}
                        s1:='Loendamise aeg: '+s1+' ms';
                        TextOut(g_DCPr,480,140,@s1[1],length(s1));
                        s1:='Kommentaar: '+g_Header3.komm;
                        TextOut(g_DCPr,50,180,@s1[1],length(s1));

                        SetTextAlign(g_DCPr,TA_CENTER+TA_TOP);
                        s1:=DateTimeToString(g_Header3.kuup);
                        TextOut(g_DCPr,480,80,@s1[1],length(s1));
                        hf3:=CreateFont(-24,0,0,0,FW_BOLD,0,0,0,0,
                                        OUT_CHARACTER_PRECIS,CLIP_CHARACTER_PRECIS,
                                        DEFAULT_QUALITY,DEFAULT_PITCH,'courier new');
                        SelectObject(g_DCPr,hf3);
                        s1:=g_Header3.nimi;
                        TextOut(g_DCPr,480,50,@s1[1],length(s1));
```

```
                        HP1:=CreatePen(PS_SOLID,2,0);
                        SelectObject(g_DCPr,HP1);
                        {SelectClipRgn(g_DCPr,0);}
                        MoveTo(g_DCPr,50,250);
                        LineTo(g_DCPr,900,250);
                        LineTo(g_DCPr,900,1000);
                        LineTo(g_DCPr,50,1000);
                        LineTo(g_DCPr,50,250);

                        SelectObject(g_DCPr,GetStockObject(BLACK_PEN));
                        hr1:=CreateRectRgn(0,300,900,1000);
                        SelectClipRgn(g_DCPr,HR1);
                        Moveto(g_DCPr,100,900);
                        LineTo(g_DCPr,850,900);
                        hf4:=CreateFont(-12,0,0,0,FW_BOLD,0,0,0,0,
                                        OUT_CHARACTER_PRECIS,CLIP_CHARACTER_PRECIS,
                                        DEFAULT_QUALITY,DEFAULT_PITCH,'courier new');
                        SelectObject(g_DCPr,hf4);
                        r1:=g_Header3.k2/18*60;
                        r2:=g_Header3.dx2/64*75;
                        l4:=round(g_Header3.dx4/64*75);
                        for i1:=0 to g_Header3.xnr do begin
                                                MoveTo(g_DCPr,100+l4*i1,900);
                                                LineTo(g_DCPr,100+l4*i1,905);
                                                str(g_Header3.x0+i1*g_Header3.dx3,s1);
                                                TextOut(g_DCPr,100+i1*l4,907,
                                                        @s1[1],length(s1));
                                                end;
                        MoveTo(g_DCPr,100,900-round(r1*m2[1,0]));
                        for i1:=1 to g_Header3.punkte
                        do begin
                           LineTo(g_DCPr,100+round(r2*i1),900-round(r1*m2[1,i1]));
                           end;
{*****}
if EndPage(g_DCPr)<0 then goto 2;
if EndDoc(g_DCPr)<0 then goto 2;
DeleteObject(hf3);
DeleteObject(hf4);
DeleteObject(HR1);
DeleteDC(g_DCPr);
exit;
2: AbortDoc(g_DCPr);
MessageBox(DlgWin,#9+'Viga printimisel'+#9,'Viga:',MB_OK);
DeleteObject(hf3);
DeleteObject(hf3);
DeleteObject(HR1);
DeleteDC(g_DCPr);
DeleteObject(HP1);*)
{***}
Prindi:=TRUE;
end;

procedure Kriips(DC: HDC; nr: integer; a: pandmed);
var i1: integer;
    x1, x2, y1, y2, deltax, deltay, t6us: real;
begin
{}
y2:=a^.k1*a^.m^[nr];
y1:=a^.k1*a^.m^[nr-1];
if ((y1>200) and (y2>200)) then exit;
deltay:=y2-y1;
x2:=a^.dx1*nr;
x1:=a^.dx1*(nr-1);
deltax:=x2-x1;
t6us:=deltay/deltax;
if abs(t6us)>1
then begin
     if t6us>0
     then for i1:=0 to round(deltay)
          do begin
             x2:=round(x1+i1/t6us);
             SetPixel(DC,20+round(x2),300-round(y1+i1),a^.v^[round(x2)]);
             if (a^.punkte<500)
             then begin
                  SetPixel(DC,21+round(x2),300-round(y1+i1),a^.v^[round(x2)]);
                  {SetPixel(DC,19+round(x2),300-round(y1+i1),a^.v^[round(x2)]);}
                  end;
```

```
            end  {for}
     else for i1:=0 downto round(deltay)
          do begin
             x2:=round(x1+i1/t6us);
             SetPixel(DC,20+round(x2),300-round(y1+i1),a^.v^[round(x2)]);
             if (a^.punkte<500)
             then begin
                  SetPixel(DC,21+round(x2),300-round(y1+i1),a^.v^[round(x2)]);
                  {SetPixel(DC,19+round(x2),300-round(y1+i1),a^.v^[round(x2)]);}
                  end;
             end;  {for}
     end
else begin  {t6us<=1}
     for i1:=0 to abs(round(deltax))
     do begin
        SetPixel(DC,20+round(x1+i1),300-round(y1+i1*t6us),a^.v^[round(x1+i1)]);
        if (a^.punkte<500)
        then begin
             {SetPixel(DC,20+round(x1+i1),301-round(y1+i1*t6us),a^.v^[round(x1+i1)]);}
             SetPixel(DC,20+round(x1+i1),299-round(y1+i1*t6us),a^.v^[round(x1+i1)]);
             end;
        end;  {for}
     end;
end;  {Kriips}

{m66tmistsykkel}
function M66daProc(DlgWin: HWnd; Msg, WParam, LParam: longint): longint; stdcall;
var i1: integer;
    i3: integer;  {WM_PAINT jaoks}
    bl1: LONGBOOL;
    r1: real;
    s1: string;
    TR1: TRect;
    PS1: TPaintStruct;
    buf: array[0..255] of char;
procedure Status(s: string);
begin
s:=s+#0;
SetDlgItemText(DlgWin,101,@s[1]);
end;  {status}
procedure StatusL(l: longint);
begin
SetDlgItemInt(DlgWin,101,l,TRUE);
end;
begin
M66daProc:=1;
case Msg
of WM_PAINT: begin
             M66daProc:=0;
             if GetUpdateRect(DlgWin,TR1,FALSE)
             then begin
                  if (g_Sodi.yks and IntersectRect(TR1,g_TRGraafik,TR1))
                  then g_Sodi.yks:=FALSE
                  end
             else if g_Sodi.yks
                  then InvalidateRect(DlgWin,@g_TRGraafik,FALSE)
                  else exit;

             BeginPaint(DlgWin,PS1);
             SetBkMode(PS1.HDC,TRANSPARENT);
             SelectObject(PS1.HDC,g_Font1);
             {MoveToEx(PS1.HDC,5,301,nil);
             LineTo(PS1.HDC,560,320);}
             {}
             if IntersectRect(TR1,g_TRxtelg,PS1.rcPaint)
             then begin
                  MoveToEx(PS1.HDC,20,301,nil);  {x-telg}
                  LineTo(PS1.HDC,550,301);
                  SetTextAlign(PS1.HDC,TA_CENTER+TA_TOP);
                  for i3:=0 to g_T66Header.xnr
                  do begin
                     MoveToEx(PS1.HDC,20+round(g_T66Header.dx5*i3),301,nil);
                     LineTo(PS1.HDC,20+round(g_T66Header.dx5*i3),304);
                     if g_M66tmiseTyyp=1
                     then begin
                          r1:=g_T66Header.x0+i3*g_T66Header.dx2;
                          s1:=L6iguReal(r1,1);
```

```
                        TextOut(PS1.HDC,20+round(i3*g_T66Header.dx5),306,@s1[1],length(s1)-1);
                        end;
                   end
              end;
         {}
         {InvalidateRect(DlgWin,nil,FALSE);}
         {InvalidateRgn(DlgWin,0,FALSE);}

         if IntersectRect(TR1,g_TRGraafik,PS1.rcPaint)
         then begin
              SelectClipRgn(PS1.HDC,g_HRGraafik);
              if g_Sodi.yks
              then begin  PS1.rcPaint:=g_TRGraafik;
                   Kriips(PS1.HDC,g_Sodi.nr,@g_T66Header);
                   g_Sodi.yks:=FALSE;
                   end
              else begin  {kogu graafik ymber joonistada}
                   FillRect(PS1.HDC,g_TRGraafik,g_HallPintsel);  {!!!!!}
                   SelectObject(PS1.HDC,g_HallSulg);
                   for i3:=1 to 10
                   do begin  {horisontaalsed kriipsud}
                      MoveToEx(PS1.HDC,20,300-18*i3,nil);
                      LineTo(PS1.HDC,550,300-18*i3);
                      end;
                   for i3:=0 to g_T66Header.xnr
                   do begin  {vertikaalsed kriipsud}
                      MoveToEx(PS1.HDC,20+round(g_T66Header.dx5*i3),120,nil);
                      LineTo(PS1.HDC,20+round(g_T66Header.dx5*i3),300);
                      end;
                   {}
                   for i3:=1 to g_Sodi.nr
                   do begin
                      Kriips(PS1.HDC,i3,@g_T66Header);
                      end;
                   end;  {else}
              SelectClipRgn(PS1.HDC,0);
              end;
         {}
         EndPaint(DlgWin,PS1);
         exit;
         end;  {WM_PAINT}
WM_COMMAND: begin
           {101 - status
           102 - tryki
           103 - korda m66tmist
           104 - sulge
           105 - faili nimi
           106 - kuupeev
           107 - algus
           108 - l6pp
           109 - kommentaar
           111 - kordsus
           113 - samm
           114 - abi
           115 - yhel lainpikkusel
           116 - pinge
           117 - takisti}
           case wParam
           of 103: begin  {Mõõda/Katkesta}
                   GetDlgItemText(DlgWin,103,@buf,10);
                   i1:=StrIComp(@buf,'Mõõda');
                   if i1=0
                   then begin  {hakkab parameetreid kontrollima}
                        {}
                        with g_T66Header
                        do begin
                           algus:=GetDlgItemInt(DlgWin,107,bl1,FALSE);
                           if ((not bl1) or (algus<200) or (algus>800))
                           then begin
                                piip;
                                Status('Spektri algus peab jääma vahemikku 200..800 nm!');
                                SendDlgItemMessage(DlgWin,107,EM_SETSEL,0,-1);
                                SetFocus(GetDlgItem(DlgWin,107));
                                exit;
                                end;
                           algus:=300*algus;
                           kordus:=GetDlgItemInt(DlgWin,111,bl1,FALSE);
```

```
if ((not bl1) or (kordus<1) or (kordus>9999))
then begin
     piip;
     Status('Kordsus peab jääma vahemikku 1..9999!');
     SendDlgItemMessage(DlgWin,111,EM_SETSEL,0,-1);
     SetFocus(GetDlgItem(DlgWin,111));
     exit;
     end;
pinge:=GetDlgItemInt(DlgWin,116,bl1,FALSE);
if ((not bl1) or (pinge<10) or (pinge>2000))
then begin
     piip;
     Status('FEK anoodpinge peab jääma vahemikku 10..2000 V!');
     SendDlgItemMessage(DlgWin,116,EM_SETSEL,0,-1);
     SetFocus(GetDlgItem(DlgWin,116));
     exit;
     end;
takisti:=1+SendDlgItemMessage(DlgWin,117,CB_GETCURSEL,0,0);
if SendDlgItemMessage(DlgWin,115,BM_GETCHECK,0,0)<>BST_CHECKED
then begin  {tavaline m66tmine}
     g_M66tmiseTyyp:=1;
     l6pp:=GetDlgItemInt(DlgWin,108,bl1,FALSE);
     if ((not bl1) or (l6pp<200) or (l6pp>800))
     then begin
          piip;
          Status('Spektri l6pp peab jääma
                 vahemikku 200..800 nm!');
          SendDlgItemMessage(DlgWin,108,EM_SETSEL,0,-1);
          SetFocus(GetDlgItem(DlgWin,108));
          exit;
          end;
     l6pp:=300*l6pp;
     if l6pp<=algus
     then begin
          piip;
          Status('Spektri lõpp peab olema algusest
                 suurem vähemalt 1 nm võrra!');
          SendDlgItemMessage(DlgWin,108,EM_SETSEL,0,-1);
          SetFocus(GetDlgItem(DlgWin,108));
          exit;
          end;
     r1:=0;
     i1:=GetDlgItemText(DlgWin,113,@buf,255);
     bl1:=TextToFloat(@buf,255,r1);
     samm:=round(300*r1);
     if ((samm<3) or (samm>300*100))
     then begin
          piip;
          Status('Skaneerimise samm peab jääma
                 vahemikku 0.01..100 nm!');
          SendDlgItemMessage(DlgWin,113,EM_SETSEL,0,-1);
          SetFocus(GetDlgItem(DlgWin,113));
          exit;
          end;
     punkte:=((l6pp-algus) div samm)+1;
     if punkte>2000
     then begin
          piip;
          Status('Katsepunkte ei tohi olla üle 2000!');
          SendDlgItemMessage(DlgWin,108,EM_SETSEL,0,-1);
          SetFocus(GetDlgItem(DlgWin,108));
          exit;
          end;
     l6pp:=algus+samm*(punkte-1);
     i1:=GetDlgItemText(DlgWin,105,@nimi,50);  {nimi}
     if (not OnNimi(nimi))
     then begin
          piip;
          Status('Faili nimi on vigane!');
          SendDlgItemMessage(DlgWin,105,EM_SETSEL,0,-1);
          SetFocus(GetDlgItem(DlgWin,105));
          exit;
          end;
     Move(g_DefaultHeader.laiend,g_T66Header.laiend,5);
     i1:=GetDlgItemText(DlgWin,109,@komm,255);  {komm}
     ArvutaParameetrid(@g_T66Header);
     end
```

```
                else begin  {yhel koordinaadil}
                     g_M66tmiseTyyp:=2;
                     SetWindowText(DlgWin,'M66tmine püsival koordinaadil');
                     SetDlgItemText(DlgWin,105,'');  {nimi}
                     SetDlgItemInt(DlgWin,108,algus div 300,FALSE);  {l6pp}
                     SetDlgItemText(DlgWin,113,'0');  {samm}
                     samm:=0;
                     punkte:=600;
                     dx1:=1;
                     xnr:=10;
                     dx5:=53;
                     end;
                end;  {with}
              {}
              Status('Oota...');
              SetDlgItemText(DlgWin,103,'Katkesta');
              EnableWindow(GetDlgItem(DlgWin,107),FALSE);  {algus}
              EnableWindow(GetDlgItem(DlgWin,108),FALSE);  {l6pp}
              EnableWindow(GetDlgItem(DlgWin,113),FALSE);  {samm}
              EnableWindow(GetDlgItem(DlgWin,105),FALSE);  {nimi}
              EnableWindow(GetDlgItem(DlgWin,109),FALSE);  {komm}
              EnableWindow(GetDlgItem(DlgWin,111),FALSE);  {kordi}
              EnableWindow(GetDlgItem(DlgWin,115),FALSE);  {yhel}
              EnableWindow(GetDlgItem(DlgWin,116),FALSE);  {pinge}
              EnableWindow(GetDlgItem(DlgWin,117),FALSE);  {takk}
              EnableMenuItem(g_hm1,204,MF_ENABLED);  {Katkesta}
              StrLCopy(@buf,g_T66Header.nimi,20);
              LStrCat(@buf,g_T66Header.laiend);
              LStrCat(@buf,': ');
              LStrCat(@buf,g_T66Header.komm);
              SetWindowText(DlgWin,@buf);
              {}
              ArvutaV2rvid(@g_T66Header);
              InvalidateRect(DlgWin,@g_TRxtelg,TRUE);
              {}
              g_M66dan:=TRUE;  {t@histab tegevust pordi kallal}
              g_Katkestan:=FALSE;
              if (AvaPort and ResetPort)
              then PostMessage(DlgWin,WM_COMMAND,505,0)  {m66tmine algab}
              else i1:=MessageBox(DlgWin,'COM-port ei avane!',
                                  'Viga:',MB_ICONEXCLAMATION+MB_OK);
              exit;
              end
         else begin  {hakkab katkestama}
              i1:=MessageBox(DlgWin,'Kas katkestan mõõtmise?',
                             'Teade:',MB_ICONEXCLAMATION+MB_YESNO);
              if i1=IDYES
              then begin
                   EnableMenuItem(g_hm1,204,MF_GRAYED);  {Katkesta}
                   StrPCopy(g_T66Header.aeg,FormatDateTime('dd.mm.yy hh:mm',Now));
                   Salvesta('mdr.tmp');
                   Status('Katkestan m66tmise...
                          (tulemused on salvestatud faili MDR.TMP');
                   EndDialog(g_hwKoordinaat,0);  {igaks juhuks}
                   g_Katkestan:=TRUE;
                   if g_Viivitus>2000
                   then begin  {tuleb j6uga katkestada}
                        if not ResetPort then PostMessage(DlgWin,WM_COMMAND,509,0);
                        end;
                   exit;
                   end
              else begin exit; end;
              end;
         end;
104:     begin  {Sulge}
         PostMessage(DlgWin,WM_CLOSE,0,0);
         exit;
         end;
102:     begin  {tryki}
         {PostMessage(DlgWin,WM_CLOSE,1,0);}
         exit;
         end;
114:     begin  {Abi}
         i1:=MessageBox(DlgWin,'Kao minema!',
                        'Abi:',MB_OK+MB_ICONINFORMATION);
         exit;
         end;
```

```
115: begin  {yhel koordinaadil}
     if SendDlgItemMessage(DlgWin,115,BM_GETCHECK,0,0)=BST_CHECKED
     then begin
          EnableWindow(GetDlgItem(DlgWin,108),FALSE);  {l6pp}
          EnableWindow(GetDlgItem(DlgWin,113),FALSE);  {samm}
          EnableWindow(GetDlgItem(DlgWin,105),FALSE);  {nimi}
          end
     else begin
          EnableWindow(GetDlgItem(DlgWin,108),TRUE);  {l6pp}
          EnableWindow(GetDlgItem(DlgWin,113),TRUE);  {samm}
          EnableWindow(GetDlgItem(DlgWin,105),TRUE);  {nimi}
          end;
     exit;
     end;
501: begin {T66Proc0 on l6ppenud}
     Status('Liigun koordinaadile...');
     g_l1:=DlgWin;  {l1 peab olema globaalne}
     g_htr1:=CreateThread(nil,0,@T66Proc4,@g_l1,0,dw1);  {koordinaadile minek}
     exit;
     end;
502: begin  {T66Proc1 saadab: yks punkt j@lle m66detud}
     inc(g_Punkt);
     if ((g_Punkt>530) and (g_M66tmiseTyyp=2)) then g_Punkt:=0;
     with g_T66Header
     do begin
        if g_m1[g_Punkt]>g_m1[max] then max:=g_Punkt;
        if k1*g_m1[g_Punkt]<=180
        then begin  {yks punkt}
             g_Sodi.nr:=g_Punkt;
             g_Sodi.yks:=TRUE;
             {RedrawWindow(DlgWin,nil,nil,RDW_INTERNALPAINT);}
             PostMessage(DlgWin,WM_PAINT,0,0);
             exit;
             end
        else begin  {vaja mastaapi muuta}
             k1:=180/g_m1[g_Punkt];
             k1max:=k1*1000;  {?????}
             k1min:=k1/5;  {?????}
             g_Sodi.nr:=g_Punkt;
             g_Sodi.yks:=FALSE;
             InvalidateRect(DlgWin,@g_TRGraafik,FALSE);
             exit;
             end;
        end;  {with}
     exit;
     end;
503: begin  {T66Proc1 l6ppenud, m66tmine valmis}
     g_l1:=DlgWin;  {l1 peab olema globaalne}
     g_htr1:=CreateThread(nil,0,@T66Proc3,@g_l1,0,dw1);  {l6petamine}
     StrPCopy(g_T66Header.aeg,FormatDateTime('dd.mm.yy hh:mm',Now));
     StrLCopy(@buf,g_DataPath,255);
     LStrCat(@buf,g_T66Header.nimi);
     LStrCat(@buf,g_T66Header.laiend);
     salvesta(@buf);
     salvesta('mdr.tmp');
     exit;
     end;
504: begin  {T66Proc3 l6pp, k6ik valmis}
     SulePort;
     g_M66dan:=FALSE;
     EnableMenuItem(g_hm1,204,MF_GRAYED);  {Katkesta}
     Status('Mõõtmine on lõppenud');
     {}
     EnableWindow(GetDlgItem(DlgWin,107),TRUE);  {algus}
     EnableWindow(GetDlgItem(DlgWin,108),TRUE);  {l6pp}
     EnableWindow(GetDlgItem(DlgWin,113),TRUE);  {samm}
     SetDlgItemText(DlgWin,105,'');
     EnableWindow(GetDlgItem(DlgWin,105),TRUE);  {nimi}
     EnableWindow(GetDlgItem(DlgWin,109),TRUE);  {komm}
     EnableWindow(GetDlgItem(DlgWin,111),TRUE);  {kordi}
     EnableWindow(GetDlgItem(DlgWin,115),TRUE);  {yhel}
     EnableWindow(GetDlgItem(DlgWin,116),TRUE);  {pinge}
     EnableWindow(GetDlgItem(DlgWin,117),TRUE);  {takk}
     CheckDlgButton(DlgWin,115,0);
     {}
     SetDlgitemText(DlgWin,103,'Mõõda');
     {EnableWindow(GetDlgItem(DlgWin,102),TRUE);}  {?????}
```

```
                    exit;
                    end;
               505: begin  {tegevuse algus}
                    g_Punkt:=-1;  {?????}
                    EnableWindow(g_hwUpDownT66,TRUE);
                    with g_T66Header  {graafiku parameetrite arvutamine}
                    do begin
                       case g_M66tmiseTyyp
                       of 1: begin  {p6hiline m66tmine}
                             end;
                          2: begin  {m66tmine pysival koordinaadil}
                             end;
                       end;  {case}
                    end;  {with}
                    g_l1:=DlgWin;  {l1 peab olema globaalne}
                    g_htr1:=CreateThread(nil,0,@T66Proc0,@g_l1,0,dw1);  {Init}
                    exit;
                    end;
               506: begin  {T66Proc4 l6pp, on koordinaadil}
                    Status('Mõõdan...');
                    g_l1:=DlgWin;  {l1 peab olema globaalne}
                    g_htr1:=CreateThread(nil,0,@T66Proc1,@g_l1,0,dw1);  {M66tmine}
                    exit;
                    end;
               507: begin  {vaba}
                    exit;
                    end;
               508: begin  {vaba}
                    exit;
                    end;
               509: begin  {jama}
                    case lParam
                    of 0: begin  {pordi jama, fataalne}
                          i1:=MessageBox(DlgWin,'COM-port ei tööta!',
                                         'Viga:',MB_OK+MB_ICONEXCLAMATION);
                          {PostMessage(DlgWin,WM_COMMAND,509,3);}
                          exit;
                          end;
                       1: begin  {PIC ei vasta}
                          i1:=MessageBox(DlgWin,'Aparatuur ei vasta!'+#13+
                                         'Vaata, et plokid oleksid sisse lülitatud!'
                                         +#13+'Kas proovin uuesti?',
                                         'Viga:',MB_RETRYCANCEL+MB_ICONEXCLAMATION);
                          if ((i1=IDRETRY){ and ResetPort})
                          then begin  {alustab uuesti}
                               PostMessage(DlgWin,WM_COMMAND,505,0);
                               exit;
                               end;
                          PostMessage(DlgWin,WM_COMMAND,509,3);  {l6pp}
                          exit;
                          end;
                       2: begin  {keris piirest v@lja}
                          i1:=MessageBox(DlgWin,'Keris piirest v@lja!'+#13
                                         +'Kas alustan uuesti?',
                                         'Viga:',MB_RETRYCANCEL+MB_ICONEXCLAMATION);
                          if ((i1=IDRETRY) and ResetPort)
                          then begin  {alustab uuesti}
                               PostMessage(DlgWin,WM_COMMAND,505,0);
                               exit;
                               end;
                          PostMessage(DlgWin,WM_COMMAND,509,3);  {l6pp}
                          exit;
                          end;
                       3: begin  {kunstlik l6petamine valmis}
                          PostMessage(DlgWin,WM_COMMAND,504,0);
                          exit;
                          end;
                    end;  {case}
                    end
               else begin M66daProc:=0; exit; end;
               end; {case wParam}
               end;  {WM_COMMAND}
WM_VSCROLL: begin  {loword(wparam)=8; lparam=aken}
            M66daProc:=0;
            if ((lParam=g_hwUpDownT66) and (loword(wParam)=8))
            then begin
                 if (hiword(wparam))>5
```

```
                    then begin  {+}
                         g_T66Header.k1:=g_T66Header.k1*1.2;
                         if g_T66Header.k1>g_T66Header.k1max
                         then g_T66Header.k1:=g_T66Header.k1max;
                         end
                    else begin  {-}
                         g_T66Header.k1:=g_T66Header.k1/1.2;
                         if g_T66Header.k1<g_T66Header.k1min
                         then g_T66Header.k1:=g_T66Header.k1min;
                         end;
                    g_T66Header.k1max:=g_T66Header.k1*1000;  {?????}
                    g_T66Header.k1min:=g_T66Header.k1/5;  {?????}
                    SendMessage(g_hwUpDownT66,UDM_SETPOS,0,5);
                    g_Sodi.yks:=FALSE;
                    InvalidateRect(DlgWin,@g_TRGraafik,TRUE);
                    end;
               end;
   WM_INITDIALOG: begin
                  SetWindowText(DlgWin,'Uue mõõtmise parameetrid:');
                  g_T66Header.m:=@g_m1;
                  g_T66Header.v:=@g_T66V2rv;
                  {g_Punkt:=0;}
                  g_M66tmiseTyyp:=1;
                  g_Sodi.nr:=0;
                  g_Sodi.yks:=FALSE;  {?????}
                  ArvutaParameetrid(@g_T66Header);
                  T2idaLahtrid(DlgWin,@g_T66Header);
                  SetDlgItemText(DlgWin,105,'');
                  SetDlgItemText(DlgWin,103,'Mõõda');
                  SetFocus(GetDlgItem(DlgWin,107));
                  EnableWindow(g_hwUpDownT66,FALSE);
                  EnableWindow(GetDlgItem(DlgWin,102),FALSE);  {tryki}
                  g_hwUpDownT66:=CreateUpDownControl(WS_CHILD+WS_VISIBLE,
                                                     500,325,10,20,DlgWin,0,
                                                     HINSTANCE,0,10,0,5);
                  EnableWindow(g_hwUpDownT66,FALSE);
                  exit;
                  end;  {WM_INITDIALOG}
   WM_CLOSE: begin
             M66daProc:=0;
             if g_M66dan
             then if MessageBox(DlgWin,'Kas katkestan mõõtmise?',
                                '',MB_ICONEXCLAMATION+MB_YESNO)<>IDYES
                  then exit;
             ResetPort;
             SulePort;
             PostMessage(g_hwMain,WM_COMMAND,501,0);
             DestroyWindow(DlgWin);
             exit;
             end
   else begin M66daProc:=0; exit; end;
end; {case Message}
end;  {M66daProc}

function AvaProc(DlgWin: HWnd; Msg, WParam, LParam: Longint): longint; stdcall;
var s1: string;
    i1, i3: integer;  {WM_PAINT jaoks}
    r1: real;
    tr1: trect;
    PS1: TPaintStruct;
procedure Status(s1: string);
begin
s1:=s1+#0;
SetDlgItemText(DlgWin,101,@s1[1]);
end;
begin
AvaProc:=1;
case Msg
of WM_PAINT: begin
             AvaProc:=0;
             if (not GetUpdateRect(DlgWin,tr1,FALSE)) then exit;
             BeginPaint(DlgWin,PS1);

             SetBkMode(PS1.HDC,TRANSPARENT);
             SelectObject(PS1.HDC,g_Font1);
             {MoveToEx(PS1.HDC,20,120,nil);
             LineTo(PS1.HDC,550,300);}
```

```
            MoveToEx(PS1.HDC,20,301,nil);  {x-telg}
            LineTo(PS1.HDC,550,301);
            SetTextAlign(PS1.HDC,TA_CENTER+TA_TOP);
            for i3:=0 to g_AvaHeader.xnr
            do begin
               MoveToEx(PS1.HDC,20+round(g_AvaHeader.dx5*i3),301,nil);
               LineTo(PS1.HDC,20+round(g_AvaHeader.dx5*i3),303);
               r1:=g_AvaHeader.x0+i3*g_AvaHeader.dx2;
               s1:=L6iguReal(r1,1);
               TextOut(PS1.HDC,20+round(i3*g_AvaHeader.dx5),305,@s1[1],length(s1)-1);
               end;
            SelectObject(PS1.HDC,g_HallSulg);
            for i3:=1 to 10 do begin
                            MoveToEx(PS1.HDC,20,300-18*i3,nil);
                            LineTo(PS1.HDC,550,300-18*i3);
                            end;
            for i3:=0 to g_AvaHeader.xnr
            do begin
               MoveToEx(PS1.HDC,20+round(g_AvaHeader.dx5*i3),120,nil);
               LineTo(PS1.HDC,20+round(g_AvaHeader.dx5*i3),300);
               end;
            SelectClipRgn(PS1.HDC,g_HRGraafik);
            for i3:=1 to g_AvaHeader.punkte-1
            do begin
               Kriips(PS1.HDC,i3,@g_AvaHeader);
               end;
            SelectClipRgn(PS1.HDC,0);
            EndPaint(DlgWin,PS1);
            exit;
            end;  {WM_PAINT}
WM_COMMAND: begin
              {101 - status
              102 - tryki
              103 - korda m66tmist
              104 - sulge
              105 - faili nimi
              106 - kuupeev
              107 - algus
              108 - l6pp
              109 - kommentaar
              111 - kordsus
              113 - samm
              114 - abi
              115 - yhel lainpikkusel
              116 - pinge
              117 - takisti}
              case wParam
              of 102: begin
                      Prindi(DlgWin,@g_AvaHeader);
                      exit;
                      end;  {Print}
                 103: begin  {Korda M66tmist}
                      {if g_M66dan}
                      if IsWindow(g_hwT66)
                      then begin  {ei saa uut m66tmist alustada}
                           i1:=MessageBox(DlgWin,'Mõõtmisaken on vaja sulgeda!',
                                          'Viga:',MB_ICONEXCLAMATION+MB_OK);
                           exit;
                           end
                      else begin  {alustab uut m66tmist}
                           {SendMessage(g_hwT66,WM_CLOSE,0,0);}
                           Move(g_AvaHeader,g_T66Header,SizeOf(andmed));
                           g_T66Header.nimi:='';
                           Move(g_DefaultHeader.laiend,g_T66Header.laiend,5);
                           PostMessage(g_hwMain,WM_COMMAND,503,0);
                           exit;
                           end;
                      end;
                 104: begin  {Sulge}
                      PostMessage(DlgWin,WM_CLOSE,0,0);
                      exit;
                      end;
                 114: begin  {abi}
                      MessageBox(DlgWin,'Kao minema!','Abi:',
                                 MB_OK+MB_ICONINFORMATION);
                      exit;
```

```
                    end
                 else begin AvaProc:=0; exit; end
              end;  {case wParam}
              end;  {WM_COMMAND}
   WM_VSCROLL: begin  {loword(wparam)=8; lparam=aken}
              AvaProc:=0;
              if ((lParam=g_hwUpDownAva) and (loword(wParam)=8))
              then begin
                   if (hiword(wparam))>5
                   then begin  {+}
                        g_AvaHeader.k1:=g_AvaHeader.k1*1.2;
                        if g_AvaHeader.k1>g_AvaHeader.k1max
                        then g_AvaHeader.k1:=g_AvaHeader.k1max;
                        end
                   else begin  {-}
                        g_AvaHeader.k1:=g_AvaHeader.k1/1.2;
                        if g_AvaHeader.k1<g_AvaHeader.k1min
                        then g_AvaHeader.k1:=g_AvaHeader.k1min;
                        end;
                   SendMessage(g_hwUpDownAva,UDM_SETPOS,0,5);
                   InvalidateRect(DlgWin,@g_TRGraafik,TRUE);
                   end;
              end;
   WM_INITDIALOG: begin  {310x215}
                  g_AvaHeader.m:=@g_m2;
                  g_AvaHeader.v:=@g_AvaV2rv;
                  ArvutaParameetrid(@g_AvaHeader);
                  T2idaLahtrid(DlgWin,@g_AvaHeader);
                  ArvutaV2rvid(@g_AvaHeader);
                  g_hwUpDownAva:=CreateUpDownControl(WS_CHILD+WS_VISIBLE,
                                                     500,325,10,20,DlgWin,0,HINSTANCE,0,
                                                     10,0,5);
                  SetWindowPos(DlgWin,0,(GetSystemMetrics(SM_CXFULLSCREEN)-620) shr 1,
                               (GetSystemMetrics(SM_CYFULLSCREEN)-430) shr 1,
                               0,0,SWP_NOSIZE+SWP_NOZORDER);
                  EnableWindow(GetDlgItem(DlgWin,105),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,107),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,108),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,109),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,111),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,113),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,115),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,116),FALSE);
                  EnableWindow(GetDlgItem(DlgWin,117),FALSE);
                  exit;
                  end;  {WM_INITDIALOG}
   WM_CLOSE: begin
             AvaProc:=0;
             PostMessage(g_hwMain,WM_COMMAND,502,0);
             DestroyWindow(DlgWin);
             exit;
             end
else begin AvaProc:=0; exit; end;
end; {case Message}
end; {AvaProc}

function MainProc(hw: HWND; Msg, wParam, lParam: longint): longint; stdcall;
var i1: integer;
    buf1, buf2: array[0..255] of char;
    s1: string;
    r1: real;
    bl1: LONGBOOL;
    p1: pointer;
begin
MainProc:=0;
case Msg
of WM_COMMAND: begin
               case wparam
               of 101: begin  {Open}
                       if OpenFileName(1,s1)
                       then begin
                            if LoeSisse(@s1[1],i1)
                            then begin
                                 EnableMenuItem(g_hm1,101,MF_GRAYED);  {Ava}
                                 {EnableMenuItem(g_hm1,102,MF_ENABLED);}  {Salvesta}
                                 EnableMenuItem(g_hm1,103,MF_ENABLED);  {Tryki}
                                 g_hwAva:=CreateDialog(hInstance,'FAIL',hw,@AvaProc);
```

```
                              exit;
                              end
                         else begin
                              str(i1,s1);
                              s1:='Fail on vigane, kood '+s1+#0;
                              i1:=MessageBox(hw,@s1[1],'Viga:',MB_OK+MB_ICONEXCLAMATION);
                              exit;
                              end;
                         end
                    else exit;
                    end;
               102: begin  {Salvesta}
                    {if OpenFileName(1,s1) then salvesta(@s1[1]);}
                    exit;
                    end;
               103: begin  {Print}
                    {print;}
                    exit;
                    end;
               104: begin  {Exit}
                    PostQuitMessage(0);
                    exit;
                    end;
               201: begin  {M66da}
                    SendMessage(g_hwT66,WM_CLOSE,0,0);  {sulgeb m66tmise akna, kui on}
                    Move(g_DefaultHeader,g_T66Header,SizeOf(andmed));
                    PostMessage(hw,WM_COMMAND,503,0);
                    exit;
                    end;
               204: begin  {Katkesta}
                    PostMessage(g_hwT66,WM_COMMAND,101,0);  {Katkesta/Sulge}
                    exit;
                    end;
               401: begin
                    MessageBox(hw,'Programm WinMDR MDR-23 juhtimiseks',
                              'Programmist',MB_OK+MB_ICONINFORMATION);
                    exit;
                    end;
               501: begin  {M66tmise aken sulgus}
                    BringWindowToTop(g_hwMain);
                    EnableMenuItem(g_hm1,201,MF_ENABLED);  {M66da}
                    exit;
                    end;
               502: begin  {faili aken sulgus}
                    EnableMenuItem(g_hm1,101,MF_ENABLED);  {Ava}
                    {EnableMenuItem(g_hm1,102,MF_GRAYED);}  {Salvesta}
                    EnableMenuItem(g_hm1,103,MF_GRAYED);  {Tryki}
                    exit;
                    end;
               503: begin  {hakkab m66tma}
                    EnableMenuItem(g_hm1,201,MF_GRAYED);  {M66da}
                    g_hwT66:=CreateDialog(hInstance,'FAIL',hw,@M66daProc);
                    exit;
                    end
               else begin MainProc:=DefWindowProc(hw,Msg,WParam,LParam); exit; end;
             end;  {case wParam}
             end;  {WM_COMMAND; g_Peab tagastama 0, kui t66tleb}
WM_CREATE: begin
           GetCurrentDirectory(255,@buf1);
           p1:=lstrcat(@buf1,'\mdr_95.ini');
           i1:=GetPrivateProfileString('GENERAL','DATAPATH','',@g_DataPath,255,p1);
           g_DataPath[i1]:='\';
           g_DataPath[i1+1]:=#0;
           with g_DefaultHeader
           do begin
              i1:=GetPrivateProfileString('GENERAL','FAILINIMI','mdr_95',@nimi,10,p1);
              i1:=GetPrivateProfileString('GENERAL','LAIEND','.dat',@laiend[1],10,p1);
              laiend[0]:='.';
              algus:=300*GetPrivateProfileInt('DEFAULT','ALGUS',400,p1);
              l6pp:=GetPrivateProfileInt('DEFAULT','L6PP',401,p1);
              i1:=GetPrivateProfileString('DEFAULT','SAMM',nil,@buf2,255,p1);
              bl1:=TextToFloat(@buf2,255,r1);
              samm:=round(300*r1);
              if ((algus<300*300) or (algus>300*800)
                  or (l6pp<(algus+300)) or (l6pp>801)
                  or (samm<30) or (samm>9999))
              then begin
```

```
                    algus:=300*400;
                    l6pp:=300*401;
                    samm:=30;
                    end;
                kordus:=GetPrivateProfileInt('DEFAULT','KORDUS',101,p1);
                pinge:=GetPrivateProfileInt('DEFAULT','PINGE',1000,p1);
                takisti:=GetPrivateProfileInt('DEFAULT','KOORMUS',1,p1);
                i1:=GetPrivateProfileString('DEFAULT','KOMMENTAAR','ff',@komm,255,p1);
                end;
            g_hm1:=GetMenu(hw);
            EnableMenuItem(g_hm1,102,MF_GRAYED);  {Salvesta}
            EnableMenuItem(g_hm1,103,MF_GRAYED);  {Tryki}
            EnableMenuItem(g_hm1,204,MF_GRAYED);  {Katkesta}
            {}
            g_HallSulg:=CreatePen(PS_SOLID,0,$009F9F9F);
            g_HallPintsel:=GetStockObject(LTGRAY_BRUSH);
            g_Font1:=GetStockObject(ANSI_VAR_FONT);
            with g_TRGraafik do begin left:=20; top:=120; right:=551; bottom:=301; end;
            g_HRGraafik:=CreateRectRgnIndirect(g_TRGraafik);
            with g_TRxtelg do begin left:=5; top:=301; right:=560; bottom:=320; end;
            {}
            g_M66tmiseTyyp:=0;
            exit;
            end;  {g_Peab tagastama 0}
   WM_DESTROY: begin
             DeleteObject(g_Font1);
             DeleteObject(g_HallPintsel);
             DeleteObject(g_HallSulg);
             DeleteObject(g_HRGraafik);
             PostQuitMessage(0);
             exit;
             end  {g_Peab tagastama 0}
   else begin MainProc:=DefWindowProc(hw,Msg,WParam,LParam); exit; end;
end;  {case Message}
end;  {MainProc}

procedure WinMain;
var Msg: TMsg;
    WindowClass: TWndClass;
begin
with WindowClass do begin
                    cbClsExtra:=0;
                    cbWndExtra:=0;
                    hIcon:=0;
                    hCursor:=LoadCursor(0,IDC_ARROW);
                    style:=0;
                    hInstance := 0;
                    lpfnWndProc:=@MainProc;
                    hbrBackground:=13;
                    lpszMenuName:='MDR_95';
                    lpszClassName:='MDR_95';
                    end;
RegisterClass(WindowClass);
g_hwMain:=CreateWindow('MDR_95','Monokromaatori MDR-23 juhtprogramm',
                      ws_OverlappedWindow,0,0,600,400,0,0,hInstance,nil);
if g_hwMain=0 then Halt(1);
ShowWindow(g_hwMain,SW_SHOWMAXIMIZED);

while GetMessage(Msg,0,0,0) do begin
                               TranslateMessage(Msg);
                               DispatchMessage(Msg);
                               end;
Halt(Msg.wParam);
end;

begin
WinMain;
end.
```

Windows-programmi ressursiskript

```
MDR_95 MENU
{
 POPUP "Fail"
 {
  MENUITEM "Ava", 101
  MENUITEM "Salvesta...", 102
  MENUITEM "Trьki", 103
  MENUITEM SEPARATOR
  MENUITEM "Vдlju", 104
 }
 POPUP "Mxxda"
 {
  MENUITEM "Mxxda", 201
  MENUITEM SEPARATOR
  MENUITEM "Katkesta", 204
 }
 POPUP "Abi"
 {
  MENUITEM "Programmist...", 401
 }
}

FAIL DIALOG 50, 25, 380, 230
STYLE WS_OVERLAPPED | WS_VISIBLE | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX
CAPTION "Mxxdetud MDR-23-ga"
FONT 8, "MS Sans Serif"
LANGUAGE LANG_NEUTRAL, SUBLANG_NEUTRAL
{
 CONTROL "Trьki", 102, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_TABSTOP, 5, 200, 60, 14
 CONTROL "Sulge", 104, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_TABSTOP, 70, 200, 60, 14
 CONTROL "Korda mxxtmist", 103, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_TABSTOP, 135, 200, 60, 14
 CONTROL "Abi", 114, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_TABSTOP, 200, 200, 60, 14
 RTEXT "Graafiku mastaap:", -1, 270, 202, 60, 10
 RTEXT "Spektri algus:", -1, 5, 6, 45, 10
 RTEXT "Spektri lxpp:", -1, 5, 21, 45, 10
 RTEXT "Samm:", -1, 5, 36, 45, 10
 RTEXT "Kommentaar:", -1, 5, 51, 45, 10
 CONTROL "444", 107, "EDIT", WS_BORDER | ES_RIGHT, 55, 5, 18, 12
 CONTROL "555", 108, "EDIT", WS_BORDER | ES_RIGHT, 55, 20, 18, 12
 CONTROL "0,09", 113, "EDIT", WS_BORDER | ES_RIGHT, 55, 35, 18, 12
 CONTROL "sdfgsrthfh  bdfghdghjdf dfshdfh", 109, "EDIT", WS_BORDER, 55, 50, 280, 12
 LTEXT "nm", -1, 75, 6, 10, 10
 LTEXT "nm", -1, 75, 21, 10, 10
 LTEXT "nm", -1, 75, 36, 10, 10
 RTEXT "Mxxtmise kordsus:", -1, 100, 6, 70, 10
 RTEXT "FEK anoodpinge:", -1, 100, 21, 70, 10
 RTEXT "FEK koormustakisti:", -1, 100, 36, 70, 10
 CONTROL "2222", 111, "EDIT", WS_BORDER | ES_RIGHT, 175, 5, 21, 12
 CONTROL "2222", 116, "EDIT", WS_BORDER | ES_RIGHT, 175, 20, 21, 12
 CONTROL "10M", 117, "COMBOBOX", CBS_DROPDOWNLIST | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         175, 35, 30, 42
 LTEXT "V", -1, 200, 21, 5, 10
 RTEXT "Faili nimi:", -1, 215, 6, 50, 10
 RTEXT "Kuupдev:", -1, 215, 21, 50, 10
 CONTROL "sdfertyu.dat", 105, "EDIT", WS_BORDER, 270, 5, 50, 12
 LTEXT "03.04.2002 12:12:00", 106, 270, 21, 70, 12
 CONTROL "Mxxda ьhel lainepikkusel", 115, "BUTTON", BS_AUTOCHECKBOX | WS_TABSTOP,
         240, 36, 100, 10
 LTEXT "Status", 101, 5, 217, 370, 12, WS_BORDER
}

KOORDINAAT DIALOG 100, 75, 160, 70
STYLE WS_POPUP | WS_CAPTION | WS_SYSMENU
CAPTION "Lainepikkus:"
FONT 8, "MS Sans Serif"
LANGUAGE LANG_NEUTRAL, SUBLANG_NEUTRAL
{
 CONTROL "Korras!", 102, "BUTTON", BS_DEFPUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         10, 46, 40, 14
 CONTROL "Tьhista", 103, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         60, 46, 40, 14
 CONTROL "Uuesti", 104, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         110, 46, 40, 14
```

```
 CONTROL "333", 101, "EDIT", ES_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_TABSTOP,
        99, 24, 17, 12
 LTEXT "nm", -1, 120, 25, 21, 8
 RTEXT "trumlilt ja kirjuta siia:", -1, 25, 25, 70, 8
 CTEXT "Vaata koordinaat monokromaatrori", -1, 20, 10, 120, 8
}
```

Windows-programmi ini-fail

```
[GENERAL]
DATAPATH=C:\PROGRA~1\WINMDR\DATA
NIMI=mdr_95
LAIEND=dat
PORT=com1
BAUD=9600

[DEFAULT]
ALGUS=400
L6PP=401
SAMM=0.1
KORDUS=100
PINGE=1000
KOORMUS=1
KOMMENTAAR=
```

Andmefaili struktuur

```
Win_MDR andmefail
26.10.04 10:08
Algus: 400 nm
Samm: 15 /300 nm
Katsepunkte: 301
Kordsus: 100
Anoodpinge: 1000 V
Koormustakisti: 1
Kommentaar: pilud 3+3 um, Hg




400.00 482
400.05 929
400.10 1033
400.15 625
400.20 526
400.25 909
400.30 994
400.35 595
400.40 541
400.45 894
400.50 964
400.55 618
400.60 532
400.65 928
400.70 1013
400.75 597
400.80 483
400.85 900
```

Linux-programm

```
program mdr1;

{$mode objfpc}

uses glib, gdk, gtk, sysutils, linux, crt;

{Define a record type to represent our app's window and controls. This
  makes it easier to mainpulate it w/o using global variables.}

type andmed = record
              algus, l6pp, kuup, takisti : longint;
              samm, kordus, pinge, punkte, max, max1, xnr, x0, dx3, dx4, nih : longint;
              k1, dx1, dx2, dx5 : real;
              x_string: array[0..10] of string[7];
              komm : array[0..255] of char;
              aeg : array[0..20] of char;
              nimi, laiend : array[0..50] of char;
              end;

type TMAINWINDOW = Record
                   Window : pGtkWindow;  {top-level window}
                   wBox0,  {peamine konteiner}
                   wNupp2, wNupp3, wNupp4, wNupp5,
                   wStatus,
                   wRaam, wGraafik,
                   wText1, wText2, wText3, wText4, wText5, wText7, wText8,
                   wTakk,  {combobox}
                   wLabel1, wLabel2, wLabel3, wLabel4, wLabel5,
                   wLabel6, wLabel7, wLabel8, wLabel9 : pGtkWidget;
                   List1: PGList;
                   Nupp1 : record
                           wNupp1 : pGtkWidget;
                           label1 : pGtkLabel;
                           glist1 : pGList;
                           end;
                   end;
     PMAINWINDOW = ^TMAINWINDOW;
     TDialog = record
               Window : PGTKDIALOG;
               wFixed : PGTKFIXED;
               wLabel: PGTKLABEL;
               wText: PGTKENTRY;
               wNupp1, wNupp2 : PGTKBUTTON;
               end;
     PDIALOG = ^TDialog;

const DTR : Cardinal=TIOCM_DTR;
      RTS : Cardinal=TIOCM_RTS;
      fL6peta : LongBool=FALSE;

var FD : longint;
    tios : Termios;
    g_s1: string;
    g_x : longint;
    buf1 : array[0..$ff] of byte;  {threadide stackid}
    g_buf1 : array[0..255] of char;
    PID01 : longint;
    g_MainWindow : TMAINWINDOW;
    g_FileDialog: PGTKWIDGET;
    g_Punkt, g_Punkt1, g_Punkt2 : longint;
    g_m1 : array[0..2000] of longint;
    g_i1 : integer;
    g_T66Header : andmed;
    g_YksPunkt : LongBool;
    g_StatusID, g_StatusLine : guint;
    g_Olukord, g_M66tmiseTyyp : byte;
    g_Dialoog : TDialog;

procedure piip; begin sound(1000); delay(10); nosound; exit; end;

procedure Status(s : string);
begin
```

```
s:=s+#0;
if g_StatusLine>0
then gtk_statusbar_remove(PGTKSTATUSBAR(g_MainWindow.wStatus),g_StatusID,g_StatusLine);
g_StatusLine:=gtk_statusbar_push(PGTKSTATUSBAR(g_MainWindow.wStatus),g_StatusID,@s[1]);
end;

procedure StatusL(l : longint);
var s : string;
begin
str(l,s);
s:=s+#0;
Status(s);
end;

procedure quit_app (widget: pGtkWidget; Window: PMAINWINDOW); cdecl;
{Clean up and shut down GTK+ signal handling.}
begin
{}
fL6peta:=TRUE;
Status('Ootan...');
if (PID01<>0) then Kill(PID01,SIGKILL);
IOCtl(FD,TIOCMBIS,@DTR);
delay(1);
IOCtl(FD,TIOCMBIC,@DTR);
delay(10);
WaitPid(PID01,nil,0);
FLock(FD,LOCK_UN);
fdClose(FD);
{}
dispose(Window);
gtk_main_quit;
end; // procedure quit_app

procedure DlgQuitProc(widget: pGtkWidget; p: pointer); cdecl;
begin
g_Olukord:=6;
end;  {DlgQuitProc1}

procedure DlgOKProc(widget: pGtkWidget; p: pointer); cdecl;
var s1: string;
    i1: integer;
    l1 : longint;
begin
s1:=StrPas(gtk_entry_get_text(g_Dialoog.wText));
val(s1,l1,i1);
if ((i1=0) and (l1>=200) and (l1<=901))
then begin
     g_x:=300*l1;
     gtk_widget_destroy(PGTKWIDGET(g_Dialoog.Window));
     end
else piip;
end;  {DlgOKProc}

function Protseduur2(Widget: PGtkWidget; event: PGdkEventExpose; Data: gpointer): longint; cdecl;
var Win: pgdkWindow;
    gc: PgdkGC;
    font: pGdkFont;
    i1: integer;
const FontName : Pchar = '-*-courier-bold-r-normal--*-90-*-*-*-*-iso8859-1';
{'-*-helvetica-bold-r-normal--*-120-*-*-*-*-iso8859-1';}
begin
result:=0;  {?????}

gc:=gdk_gc_new(widget^.Window);
Win:=widget^.window;
{With Event^.area do gdk_window_clear_area(win,x,y,width,height);}

{--------------}
(*                            if g_Punkt>0  {?????}
                           then begin
                                SelectClipRgn(g_DCT66,g_HRT66);
                                r1:=g_T66Header.k1*g_m1[0];
                                if r1>30000 then r1:=1000;
                                MoveToEx(g_DCT66,30,220-round(r1),nil);
                                for i1:=1 to g_Punkt
                                do begin
                                   r1:=g_T66Header.k1*g_m1[i1];
```

```
                                    if r1>30000 then r1:=1000;
                                    LineTo(g_DCT66,30+round(g_T66Header.dx1*i1),220-round(r1));
                                    end;
                                 SelectClipRgn(g_DCT66,0);
                                 end;
                              end;  {else}
                        end;  {case}*)
{--------------}

if g_YksPunkt
then begin
     gdk_gc_set_line_attributes(gc,1,GDK_LINE_SOLID,GDK_CAP_ROUND,GDK_JOIN_MITER);
     while (g_Punkt2<g_Punkt1)
     do begin
        inc(g_Punkt2);  {joonistatud punktide arv}
        gdk_draw_line(win,gc,10+round(g_T66Header.dx1*(g_Punkt2-1)),
                      round(290-g_T66Header.k1*g_m1[g_Punkt2-1]),
                      10+round(g_T66Header.dx1*g_Punkt2),round(290-
g_T66Header.k1*g_m1[g_Punkt2]));
        end;
     g_YksPunkt:=FALSE;
     end
else begin  {k6ik vaja yle joonistada}
     gdk_window_clear_area(win,10,0,640,{289}310);
     gdk_gc_set_line_attributes(gc,3,GDK_LINE_SOLID,GDK_CAP_ROUND,GDK_JOIN_MITER);
     gdk_draw_line(win,gc,10,290,640,290);
     gdk_gc_set_line_attributes(gc,1,GDK_LINE_SOLID,GDK_CAP_ROUND,GDK_JOIN_MITER);
     font:=gdk_font_load(FontName);
     {}
     for i1:=0 to g_T66Header.xnr
     do begin
        gdk_draw_line(win,gc,10+round(g_T66Header.dx5*i1),290,10+round(g_T66Header.dx5*i1),295);
        gdk_draw_string(win,font,gc,round(i1*g_T66Header.dx5),310,@g_T66Header.x_string[i1][1]);
        end;
     {}
     for g_Punkt2:=1 to g_Punkt1
     do gdk_draw_line(win,gc,10+round(g_T66Header.dx1*(g_Punkt2-1)),
                      round(290-g_T66Header.k1*g_m1[g_Punkt2-1]),
                      10+round(g_T66Header.dx1*g_Punkt2),
                      round(290-g_T66Header.k1*g_m1[g_Punkt2]));
     end;
result:=0;
end;  {Protseduur2}

procedure KontrollAeg(l: longint);
begin
TCGetAttr(FD,tios);
tios.c_cc[VTIME]:=l;
TCSetAttr(FD,TCSANOW,tios);

end;

function T66Proc0(p: pointer): longint; cdecl;  {Init ja reeperi otsimine}
var i1: integer;
    l1: longint;
    buf: array[0..20] of char;
label 1;
begin
buf[0]:='n';  {takisti}
buf[1]:=char($20);  {!!!!!}
(*case g_T66Header.takisti of 2: buf[1]:=char($20);  {!!!!!}
                              3: buf[1]:=char($20);
end;  {case}*)
IOCtl(FD,TIOCMBIS,@DTR);
delay(1);
IOCtl(FD,TIOCMBIC,@DTR);
delay(10);
TCFlush(FD,TCIOFLUSH);
fdWrite(FD,buf,2);
KontrollAeg(20);
l1:=fdRead(FD,buf,1); if l1<>1 then goto 1;
buf[0]:='k';  {k6rgepinge}
l1:=round(1.53*g_T66Header.pinge);
Move(l1,buf[1],2);
fdWrite(FD,buf,3);
fdRead(FD,buf,1);
{}
```

```
buf[0]:='p';  {samm paremale}  {liikumine reeperile}
buf[1]:=#1;  {lo}
buf[2]:=#0;  {hi}
while (ord(buf[10]) and 8) = 0
do begin
   fdWrite(FD,buf,3);
   fdRead(FD,buf[10],1);
   end;
while (ord(buf[10]) and 8) <> 0
do begin
   fdWrite(FD,buf,3);
   fdRead(FD,buf[10],1);
   end;
buf[1]:=#60;  {lo}
buf[2]:=#0;  {hi}
fdWrite(FD,buf,3);
fdRead(FD,buf[10],1);
{}
gdk_threads_enter;
g_Olukord:=3;
gdk_threads_leave;
Result:=0;
exit;
1:
gdk_threads_enter;
g_Olukord:=12;
gdk_threads_leave;
Result:=0;
exit;
end;  {T66Proc0}

function T66Proc1(p: pointer): longint; cdecl;  {koordinaadile minek ja m66tmine}
var i1: integer;
    l1: LONGINT;
    buf: array[0..20] of char;
begin
l1:=g_T66Header.algus-g_x;
if l1>0 then buf[0]:='v' else buf[0]:='p';
buf[1]:=#255;
buf[2]:=#255;
for i1:=1 to (abs(l1) div 65535)
do begin
   fdWrite(FD,buf,3);
   fdRead(FD,buf[10],1);
   if (l1<0) then g_x:=g_x-65535 else g_x:=g_x+65535;
   delay(1000);  {!!!!!}
   end;
if l1>0 then buf[0]:='v' else buf[0]:='p';
l1:=abs(g_T66Header.algus-g_x);
buf[1]:=char(l1 mod 256);
buf[2]:=char(l1 div 256);
fdWrite(FD,buf,3);
fdRead(FD,buf[10],1);
g_x:=g_T66Header.algus;
{}
{if g_M66tmiseTyyp=2 then faas_maha  !!!!!}
{}
for i1:=0 to g_T66Header.punkte do g_m1[i1]:=0;
g_Olukord:=8;  {j6udis koordinaadile}
g_Punkt:=0;  {m66tmine}
buf[0]:='m';
buf[1]:='v';  {n sammu vasakule}
Move(g_T66Header.samm,buf[2],2);
for i1:=1 to g_T66Header.kordus  {esimene punkt}
do begin
   fdWrite(FD,buf,1);
   fdRead(FD,l1,3);
   g_m1[0]:=g_m1[0]+(l1 and $0000ffff);
   end;
if g_m1[0]<>0 then g_T66Header.k1:=270/g_m1[0] else g_T66Header.k1:=1;
g_T66Header.max:=0;
g_T66Header.max1:=-1;
repeat  {ylej@@nud punktid}
fdWrite(FD,buf[1],3);
fdRead(FD,buf[10],1);
inc(g_x,g_T66Header.samm);
{if ((punkt>560) and (g_M66tmiseTyyp=2)) then punkt:=0;}  {!!!!!}
```

```
for i1:=1 to g_T66Header.kordus
do begin  {m66tmine ja keskmistamine}
   fdWrite(FD,buf,1);
   fdRead(FD,l1,3);
   g_m1[g_Punkt+1]:=g_m1[g_Punkt+1]+(l1 and $0000ffff);
   end;
if (g_m1[g_Punkt]>g_m1[g_T66Header.max])
then begin
     g_T66Header.max:=g_Punkt;
     {g_T66Header.k1:=270/g_m1[g_T66Header.max]}
     end;
inc(g_Punkt);
until g_Punkt=g_T66Header.punkte-1;
{}
buf[0]:='k';  {pinge maha}
buf[1]:=#0;
buf[2]:=#0;
fdWrite(FD,buf,3);
fdRead(FD,buf[10],1);
buf[0]:='f';  {faas maha}
fdWrite(FD,buf,1);
fdRead(FD,buf[10],1);
g_Olukord:=11;  {valmis}
Result:=0;
end;  {T66Proc1}

function StrToLong(var s : string; var l : longint) : LongBool;  {leiab stringist t@isarvu}
var i, j : integer;
begin
StrToLong:=FALSE;
i:=1;
while (not (s[i] in ['0'..'9'])) do begin s[i]:=#32; inc(i); end;
j:=i;
while (s[i] in ['0'..'9']) do inc(i);
s:=copy(s,j,i-j);
val(s,l,i);
if i=0 then StrToLong:=TRUE;
end;

function ArvutaParameetrid: LONGBOOL;
var i1: integer;
    r1: real;
begin
Result:=FALSE;
with g_T66Header  {graafiku parameetrite arvutamine}
do begin
   dx1:=630/(punkte-1);  {x-nihe pikslites}
   r1:=samm*(punkte-1)/300;
   dx5:=630/r1;  {pikslit nanomeetrile}
   r1:=ln(r1)/ln(10);  {log(l6pp-algus)}
   if r1>=0 then r1:=trunc(r1) else r1:=trunc(r1)-1;  {ceil ei t66ta}
   dx2:=round(exp(ln(10)*r1));  {jaotise suurus}
   xnr:=trunc((l6pp-algus)/(300*dx2));  {jaotiste arv}
   if xnr<2 then dx2:=dx2/5 else if xnr<5 then dx2:=dx2/2;
   r1:=samm*(punkte-1)/(300*dx2);
   xnr:=round(r1);  {jaotiste arv}
   dx5:=dx5*dx2;
   x0:=round(algus/300);
   for i1:=0 to xnr
   do begin
      r1:=x0+i1*dx2;
      if (trunc(r1)=r1) then str(trunc(r1),x_string[i1]) else str(r1:1:1,x_string[i1]);
      x_string[i1]:=x_string[i1]+#0;
      end;
   end;  {with}
Result:=TRUE;
end;

function LoeSisse(nimi: pChar): LONGBOOL; cdecl;
var i1 : integer;
    f1 : text;
    s1 : string;
    bl : LONGBOOL;
    r1 : real;
label 1;
begin
LoeSisse:=FALSE;
```

```
assign(f1,nimi);
reset(f1);
with g_T66Header
do begin
   readln(f1,s1);
   if (s1<>'Lin_MDR andmefail') then exit;
   readln(f1,aeg);
   readln(f1,s1);
   bl:=StrToLong(s1,algus);
   if ((algus<200) or (algus>800) or (not bl)) then goto 1;
   algus:=300*algus;
   readln(f1,s1);
   bl:=StrToLong(s1,samm);
   if ((samm<3) or (samm>9999) or (not bl)) then goto 1;
   readln(f1,s1);
   bl:=StrToLong(s1,punkte);
   if ((punkte<3) or (punkte>9999) or (not bl)) then goto 1;
   l6pp:=algus+(punkte-1)*samm;
   readln(f1,s1);
   bl:=StrToLong(s1,kordus);
   if ((kordus<10) or (kordus>9999) or (not bl)) then goto 1;
   readln(f1,s1);
   bl:=StrToLong(s1,pinge);
   if ((pinge<0) or (pinge>2000) or (not bl)) then goto 1;
   readln(f1,s1);
   bl:=StrToLong(s1,takisti);
   if ((takisti<0) or (takisti>3) or (not bl)) then goto 1;
   readln(f1,s1);  {13}
   StrPCopy(komm,copy(s1,13,255));
   for i1:=1 to 7 do readln(f1);
   max:=0;
   for i1:=0 to punkte-1 do begin
                            readln(f1,r1,g_m1[i1]);
                            if g_m1[i1]>g_m1[max] then max:=i1;
                            end;
   if g_m1[max]<>0 then k1:=250/g_m1[max] else k1:=1;
   end;  {with}
Close(f1);
LoeSisse:=TRUE;
exit;
1: Close(f1);
end;

procedure FileOKProc(widget: pGtkWidget; p: pointer); cdecl;
var buf: array[0..100] of char;
    s1: string;
    r1: real;
begin
StrLCopy(buf,gtk_file_selection_get_filename(GTK_FILE_SELECTION(g_FileDialog)),100);
gtk_widget_destroy(g_FileDialog);

if LoeSisse(buf)
then begin
     with g_T66Header
     do begin
        str(algus div 300,s1); s1:=s1+#0;
gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText1),@s1[1]);
        str(l6pp div 300,s1); s1:=s1+#0;
gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText2),@s1[1]);
        r1:=samm/300;
        str(r1:1:1,s1); s1:=s1+#0; gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText3),@s1[1]);
        gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText4),komm);
        str(pinge,s1); s1:=s1+#0; gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText5),@s1[1]);
        case takisti
        of 2: buf:='3M';
           3: buf:='10M'
           else buf:='1M';
        end;  {case}
        gtk_entry_set_text(PGTKENTRY(GTK_COMBO(g_MainWindow.wTakk)^.entry),buf);
        str(kordus,s1); s1:=s1+#0; gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText7),@s1[1]);
        g_Punkt1:=punkte-1;
        end;  {with}
     ArvutaParameetrid;
     gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText8),'');
     g_YksPunkt:=FALSE;
     gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.wGraafik),'expose_event');
     end;
```

```
end;

function AvaProc(p : pointer) : longint; cdecl;
begin
Result:=0;
if g_Olukord<>0 then exit;

g_FileDialog:=gtk_file_selection_new('Vali fail:');
gtk_window_set_modal(PGTKWINDOW(g_FileDialog),TRUE);
gtk_window_set_policy(PGTKWINDOW(g_FileDialog),0,0,0);
gtk_window_set_position(PGTKWINDOW(g_FileDialog),GTK_WIN_POS_CENTER);
gtk_signal_connect(GTK_OBJECT(GTK_FILE_SELECTION(g_FileDialog)^.ok_button),'clicked',
                   GTK_SIGNAL_FUNC(@FileOKProc),nil);
gtk_signal_connect_object(GTK_OBJECT(GTK_FILE_SELECTION(g_FileDialog)^.cancel_button),'clicked',
                          GTK_SIGNAL_FUNC(@gtk_widget_destroy),PGTKOBJECT(g_FileDialog));
{gtk_file_selection_set_filename(GTK_FILE_SELECTION(g_FileDialog),'/home/tudeng/');}
gtk_widget_show(g_FileDialog);
end;

function Salvesta(nimi : pChar) : boolean;
var f : text;
    i : integer;
    r : real;
begin
Salvesta:=FALSE;
Assign(f,'./'+StrPas(nimi));  {kataloog  !!!!!}
ReWrite(f);
with g_T66Header
do begin
   writeln(f,'Lin_MDR andmefail');
   writeln(f,aeg);
   writeln(f,'Algus: ',algus div 300,' nm');
   writeln(f,'Samm: ',samm,' /300 nm');
   writeln(f,'Katsepunkte: ',punkte);
   writeln(f,'Kordsus: ',kordus);
   writeln(f,'Anoodpinge: ',pinge,' V');
   writeln(f,'Koormustakisti: ',takisti);
   writeln(f,'Kommentaar: ',komm);
   for i:=1 to 7 do writeln(f);
   for i:=0 to punkte-1 do begin
                           r:=(algus div 300)+i*samm/300;
                           writeln(f,r:2:2,' ',g_m1[i]);
                           end;
   end;  {with}
Close(f);
end;  {salvesta}

function KellProc(p: pointer): longint; cdecl;
begin
Result:=1;
case g_Olukord
of 0: begin  {midagi ei toimu}
      exit;
      end;
   1: begin  {initsialiseerimine ja reeperi otsimine}
      exit;
      end;
   3: begin  {init tehtud}
      WaitPid(PID01,nil,0);
      Status('Vaata lainepikkus monokromaatori esipaneelilt ja sisesta siia!');
      g_Olukord:=5;
      with g_Dialoog
      do begin
         g_x:=0;
         Window:=pGtkDialog(gtk_dialog_new);
         gtk_window_set_title(PGTKWINDOW(Window),'Koordinaat');
         gtk_window_set_modal(PGTKWINDOW(Window),TRUE);
         gtk_window_set_policy(PGTKWINDOW(Window),0,0,0);
         gtk_widget_set_usize(PGTKWIDGET(Window),200,120);
         gtk_window_set_transient_for(PGTKWINDOW(Window),PGTKWINDOW(g_MainWindow.Window));
         gtk_window_set_position(PGTKWINDOW(Window),GTK_WIN_POS_CENTER);
         wFixed:=PGTKFIXED(gtk_fixed_new);
         gtk_container_add(GTK_CONTAINER(Window^.vbox),PGTKWIDGET(wFixed));
         wNupp1:=PGTKBUTTON(gtk_button_new_with_label('OK'));
         gtk_box_pack_start(PGTKBOX(Window^.action_area),PGTKWIDGET(wNupp1),FALSE,FALSE,5);
         gtk_widget_set_usize(PGTKWIDGET(wNupp1),80,35);  {70,25}
;;;
```

```
         gtk_widget_set_flags(PGTKWIDGET(wNupp1),GTK_CAN_DEFAULT);  {!!!!!}
         gtk_window_set_default(PGTKWINDOW(Window),PGTKWIDGET(wNupp1));
;;;
         wNupp2:=PGTKBUTTON(gtk_button_new_with_label('Cancel'));
         gtk_box_pack_start(PGTKBOX(Window^.action_area),PGTKWIDGET(wNupp2),FALSE,FALSE,5);
         gtk_widget_set_usize(PGTKWIDGET(wNupp2),70,25);
         gtk_signal_connect(pGtkObject(Window),'destroy',GTK_SIGNAL_FUNC(@DlgQuitProc),nil);
         gtk_signal_connect(pGtkObject(wNupp1), 'clicked',GTK_SIGNAL_FUNC(@DlgOKProc),nil);
         gtk_signal_connect_object(pGtkObject(wNupp2), 'clicked',
                                   GTK_SIGNAL_FUNC(@gtk_widget_destroy),
                                   PGTKOBJECT(Window));
         wLabel:=PGTKLABEL(gtk_label_new('Sisesta koordinaat:'));
         {gtk_widget_set_usize(PGTKWIDGET(wLabel),100,20);}
         gtk_fixed_put(wFixed,PGTKWIDGET(wLabel),20,15);
         wText:=PGTKENTRY(gtk_entry_new);
         gtk_widget_set_usize(PGTKWIDGET(wText),35,25);
         gtk_fixed_put(wFixed,pGtkWidget(wTEXT),80,40);
         gtk_entry_set_max_length(wText,3);
         gtk_entry_set_text(wText,'444');
         gtk_window_set_focus(PGTKWINDOW(Window),PGTKWIDGET(wText));
         gtk_editable_select_region(PGTKEDITABLE(wText),0,5);
         gtk_widget_show_all(PGtkWidget(Window));
         end;  {with}
      exit;
      end;
   5: begin  {koordinaadi dialoog on ees}
      exit;
      end;
   6: begin  {koordinaat on sisestatud}
      if ((g_x>=200*300) and (g_x<=901*300))
      then begin  {k6ik korras}
           Status('Liigun koordinaadile... Katkestamiseks pressi "Stopp"');
           g_Olukord:=7;
           g_Punkt:=0;
           g_Punkt1:=0;
           g_Punkt2:=0;
           fL6peta:=FALSE;
           PID01:=Clone(@T66Proc1,@buf1[$ff],
                        CLONE_VM+CLONE_FS+CLONE_SIGHAND+CLONE_FILES+SIGCHLD,nil);
           exit;
           end
      else begin  {katkestamine}
           g_Olukord:=0;
           gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.Nupp1.wNupp1),'clicked');
           exit;
           end;
      end;
   7: begin  {liikumine koordinaadile}
      exit;
      end;
   8: begin  {J6udis koordinaadile}
      Status('M66dan spektrit... Katkestamiseks pressi "Stopp"');
      g_Punkt1:=0;
      {g_YksPunkt:=FALSE;
      gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.wGraafik),'expose_event');}
      g_Olukord:=9;
      exit;
      end;
   9: begin  {m66tmine ja sammumine}
      gdk_threads_enter;  {?????}
      if (g_Punkt>g_Punkt1)
      then begin
           g_Punkt1:=g_Punkt;
           if (g_T66Header.max1=g_T66Header.max)
           then g_YksPunkt:=TRUE
           else begin
                g_T66Header.max1:=g_T66Header.max;
                g_T66Header.k1:=270/g_m1[g_T66Header.max1];
                end;
           gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.wGraafik),'expose_event');
           end;
      gdk_threads_leave;  {?????}
      exit;
      end;
  11: begin  {valmis}
       g_Punkt1:=g_Punkt;
       g_YksPunkt:=TRUE;
```

```
        gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.wGraafik),'expose_event');
        g_Olukord:=0;
        Salvesta(g_T66Header.nimi);
        gtk_entry_set_text(PGTKENTRY(g_MainWindow.wText8),'');
        Status('M66tmine on l6ppenud');
        gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.Nupp1.wNupp1),'clicked');
        exit
        end;
    12: begin  {sidekatkestus}
        Status('Paha');
        gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.Nupp1.wNupp1),'clicked');
        exit;
        end
    else begin  {?????}
         exit;
         end
end;  {case}

end;

function OnNimi(p : pChar) : boolean;
var i : integer;
    buf : array[0..50] of char;
begin
OnNimi:=FALSE;
StrLCopy(buf,p,50);
if StrLen(buf)=0 then exit;
for i:=0 to StrLen(buf)-1 do if (not (buf[i] in ['A'..'Z','a'..'z','0'..'9','_','-',' ']))
                             then exit;
OnNimi:=TRUE;
end;

function AllaProc(p : pointer) : longint; cdecl;  {graafiku mastaabi muutmine}
begin
Result:=1;
g_T66Header.k1:=g_T66Header.k1/1.2;
gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.wGraafik),'expose_event');
end;  {AllaProc}

function YlesProc(p : pointer) : longint; cdecl;  {graafiku mastaabi muutmine}
begin
Result:=1;
g_T66Header.k1:=1.2*g_T66Header.k1;
gtk_signal_emit_by_name(GTK_OBJECT(g_MainWindow.wGraafik),'expose_event');
end;  {YlesProc}

function StartStopProc(p : pointer) : longint; cdecl;  {Start/Stop}
var pChar1 : pChar;
    s1 : string;
    i1 : integer;
    wi1 : PGTKWIDGET;
    r1 : real;
Label 1;
begin
Result:=1;
gtk_label_get(GTK_LABEL(g_MainWindow.Nupp1.label1),@pChar1);
if (pChar1 = 'Start')
then begin  {hakkab m66tma}
     with g_T66Header
     do begin  {parameetrite kontrollimine}
        pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText1));
        s1:=StrPas(pChar1);
        val(s1,algus,i1);
        if ((algus<200) or (algus>800) or (i1<>0))
        then begin
             wi1:=g_MainWindow.wText1;
             s1:='Skaneerimise algus peab j@@ma vahemikku [200..800] nm';
             goto 1;
             end;
        algus:=300*algus;
        pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText7));
        s1:=StrPas(pChar1);
        val(s1,kordus,i1);
        if ((kordus<10) or (kordus>9999) or (i1<>0))
        then begin
             wi1:=g_MainWindow.wText7;
             s1:='M66tmise kordsus peab j@@ma vahemikku [10..9999]';
```

```
          goto 1;
          end;
     pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText5));
     s1:=StrPas(pChar1);
     val(s1,pinge,i1);
     if ((pinge<10) or (pinge>2000) or (i1<>0))
     then begin
          wi1:=g_MainWindow.wText2;
          s1:='FEK anoodpinge peab j@@ma vahemikku [10..2000] V';
          goto 1;
          end;
     pChar1:=gtk_entry_get_text(GTK_ENTRY(GTK_COMBO(g_MainWindow.wTakk)^.entry));
     s1:=StrPas(pChar1);
     takisti:=1;
     if s1='3M' then takisti:=2 else if s1='10M' then takisti:=3;
     {if SendDlgItemMessage(DlgWin,115,BM_GETCHECK,0,0)<>BST_CHECKED  !!!!!}
     g_M66tmiseTyyp:=1;
     if g_M66tmiseTyyp=1
     then begin  {tavaline m66tmine}
          pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText2));
          s1:=StrPas(pChar1);
          val(s1,l6pp,i1);
          l6pp:=300*l6pp;
          if ((l6pp<=algus) or (l6pp>801*300) or (i1<>0))
          then begin
               wi1:=g_MainWindow.wText2;
               s1:='Skaneerimise l6pp peab j@@ma vahemikku (Algus..801] nm';
               goto 1;
               end;
          pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText3));
          s1:=StrPas(pChar1);
          val(s1,r1,i1);
          samm:=round(300*r1);
          if ((samm<3) or (samm>=(l6pp-algus)) or (i1<>0))
          then begin
               wi1:=g_MainWindow.wText3;
               s1:='Skaneerimise samm peab j@@ma vahemikku [0.1..(l6pp-algus)) nm';
               goto 1;
               end;
          punkte:=((l6pp-algus) div samm)+1;
          if punkte>2000
          then begin
               wi1:=g_MainWindow.wText2;
               s1:='Katsepunktide arv ei tohi olla suurem kui 2000';
               goto 1;
               end;
          l6pp:=algus+samm*(punkte-1);
          pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText8));
          StrLCopy(nimi,pChar1,50);
          if (not OnNimi(nimi))
          then begin
               wi1:=g_MainWindow.wText8;
               s1:='Failinimi on vigane!';
               goto 1;
               end;
          pChar1:=gtk_entry_get_text(PGTKENTRY(g_MainWindow.wText4));
          StrLCopy(komm,pChar1,50);
          end
     else begin  {yhel koordinaadil}
          g_M66tmiseTyyp:=2;
          {!!!!!!}
          end;
     end;  {with}
ArvutaParameetrid;
{}
{gtk_entry_set_editable(PGTKENTRY(g_MainWindow.wText1),FALSE);}  {hall on ilusam}
gtk_widget_set_sensitive(g_MainWindow.wText1,FALSE);
gtk_widget_set_sensitive(g_MainWindow.wText2,FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText3),FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText4),FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText5),FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText7),FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText8),FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wTakk),FALSE);
gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wNupp2),FALSE);

gtk_label_set_text(PGTKLABEL(g_MainWindow.Nupp1.label1),'Stopp');
```

```
     {}
     Status('Initsialiseerin...');
     fL6peta:=FALSE;
     g_Olukord:=1;
     PID01:=Clone(@T66Proc0,@buf1[$ff],CLONE_VM+CLONE_FS+CLONE_SIGHAND+CLONE_FILES+SIGCHLD,nil);
     {}
     exit;
     end
else begin  {hakkab katkestama}
     fL6peta:=TRUE;
     Kill(PID01,SIGKILL);
     Salvesta('./mdr.tmp');
     {}
     IOCtl(FD,TIOCMBIS,@DTR);
     delay(1);
     IOCtl(FD,TIOCMBIC,@DTR);
     delay(10);
     {}
     WaitPid(PID01,nil,0);
     gtk_widget_set_sensitive(g_MainWindow.wText1,TRUE);
     gtk_widget_set_sensitive(g_MainWindow.wText2,TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText3),TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText4),TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText5),TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText7),TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wText8),TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wTakk),TRUE);
     gtk_widget_set_sensitive(PGTKWIDGET(g_MainWindow.wNupp2),TRUE);
     gtk_label_set_text(PGTKLABEL(g_MainWindow.Nupp1.label1),'Start');
     {Status('Sisesta parameetrid ja m66da');}
     g_Olukord:=0;
     exit;
     end;
1: Status(s1);
gtk_editable_select_region(PGTKEDITABLE(wi1),0,10);
exit;
end;  {StartStopProc}

function HelpProc(p : pointer) : longint; cdecl;  {Help}
begin
Result:=1;
{}
with g_Dialoog
do begin
   Window:=pGtkDialog(gtk_dialog_new);
   gtk_window_set_title(PGTKWINDOW(Window),'Programmist:');
   gtk_window_set_modal(PGTKWINDOW(Window),TRUE);
   gtk_window_set_policy(PGTKWINDOW(Window),0,0,0);
   gtk_widget_set_usize(PGTKWIDGET(Window),300,120);
   gtk_window_set_transient_for(PGTKWINDOW(Window),PGTKWINDOW(g_MainWindow.Window));
   gtk_window_set_position(PGTKWINDOW(Window),GTK_WIN_POS_CENTER);
   wFixed:=PGTKFIXED(gtk_fixed_new);
   gtk_container_add(GTK_CONTAINER(Window^.vbox),PGTKWIDGET(wFixed));
   wNupp1:=PGTKBUTTON(gtk_button_new_with_label('OK'));
   gtk_box_pack_start(PGTKBOX(Window^.action_area),PGTKWIDGET(wNupp1),FALSE,FALSE,5);
   gtk_widget_set_usize(PGTKWIDGET(wNupp1),80,35);  {70,25}
   gtk_widget_set_flags(PGTKWIDGET(wNupp1),GTK_CAN_DEFAULT);
   gtk_window_set_default(PGTKWINDOW(Window),PGTKWIDGET(wNupp1));
   {gtk_signal_connect(pGtkObject(Window),'destroy',GTK_SIGNAL_FUNC(@DlgQuitProc),nil);}
   gtk_signal_connect_object(pGtkObject(wNupp1), 'clicked',GTK_SIGNAL_FUNC(@gtk_widget_destroy),
                             PGTKOBJECT(Window));
   wLabel:=PGTKLABEL(gtk_label_new('Programm MDR-23 juhtimiseks'));
   gtk_fixed_put(wFixed,PGTKWIDGET(wLabel),30,30);
   gtk_widget_show_all(PGtkWidget(Window));
   end;  {with}
{}
end;  {HelpProc}

begin
{}
g_Olukord:=0;
PID01:=0;
g_Punkt:=0;
g_Punkt1:=0;
g_Punkt2:=0;
g_YksPunkt:=FALSE;
g_T66Header.xnr:=0;
```

```
for g_i1:=0 to 100 do g_m1[g_i1]:=random(100);
{}
gtk_init(@argc, @argv);
with g_MainWindow
do begin
   Window:=pGtkWindow(gtk_window_new(GTK_WINDOW_TOPLEVEL));
   gtk_window_set_policy(Window,0,0,1);  { 001 ?????}
   gtk_window_set_title(Window,'MDR-23 juhtimine');
   gtk_widget_set_usize(PGTKWIDGET(Window),700,500);
   gtk_window_set_position(PGTKWINDOW(Window),GTK_WIN_POS_CENTER);  {?????}
   gtk_signal_connect(pGtkObject(Window),'destroy',GTK_SIGNAL_FUNC(@quit_app),nil);
   wBox0:=gtk_fixed_new;
   gtk_container_add(GTK_CONTAINER(Window),wBox0);

   wRaam:=gtk_frame_new('');
   gtk_widget_set_usize(wRaam,680,340);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wRaam),10,120);
   wGraafik:=gtk_drawing_area_new;
   gtk_signal_connect(GTK_OBJECT(wGraafik),'expose_event',GTK_SIGNAL_FUNC(@Protseduur2),
                      {@g_MainWindow}nil);
   gtk_drawing_area_size(PGTKDRAWINGAREA(wGraafik),660,310);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wGraafik),20,140);
   wStatus:=gtk_statusbar_new;
   gtk_widget_set_usize(wStatus,694,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wStatus),3,474);
   g_StatusLine:=0;
   g_StatusID:=gtk_statusbar_get_context_id(PGTKSTATUSBAR(wStatus),'Statusbar');

   wLabel1:=gtk_label_new('Spektri algus (nm):'); gtk_widget_set_usize(wLabel1,140,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel1),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel1),10,10);
   wLabel2:=gtk_label_new('Spektri l6pp (nm):'); gtk_widget_set_usize(wLabel2,140,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel2),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel2),10,35);
   wLabel3:=gtk_label_new('Samm (nm):'); gtk_widget_set_usize(wLabel3,140,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel3),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel3),10,60);
   wLabel4:=gtk_label_new('Kommentaar:'); gtk_widget_set_usize(wLabel4,140,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel4),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel4),10,85);

   wText1:=gtk_entry_new; gtk_widget_set_usize(wText1,35,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText1),155,10);
   gtk_entry_set_max_length(PGTKENTRY(wText1),3);
   wText2:=gtk_entry_new; gtk_widget_set_usize(wText2,35,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText2),155,35);
   gtk_entry_set_max_length(PGTKENTRY(wText2),3);
   wText3:=gtk_entry_new; gtk_widget_set_usize(wText3,40,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText3),155,60);
   gtk_entry_set_max_length(PGTKENTRY(wText3),4);
   wText4:=gtk_entry_new; gtk_widget_set_usize(wText4,410,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText4),155,85);

   wLabel5:=gtk_label_new('FEK anoodpinge:'); gtk_widget_set_usize(wLabel5,130,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel5),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel5),210,10);
   wLabel6:=gtk_label_new('Koormustakisti:'); gtk_widget_set_usize(wLabel6,130,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel6),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel6),210,35);
   wLabel7:=gtk_label_new('Keskmistada:'); gtk_widget_set_usize(wLabel7,130,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel7),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel7),210,60);
   wLabel8:=gtk_label_new('Failinimi:'); gtk_widget_set_usize(wLabel8,100,25);
   gtk_misc_set_alignment(GTK_MISC(wLabel8),1,0.5);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wLabel8),390,10);

   wText5:=gtk_entry_new; gtk_widget_set_usize(wText5,45,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText5),345,10);
   gtk_entry_set_max_length(PGTKENTRY(wText5),4);
   wTakk:=gtk_combo_new;
   gtk_editable_set_editable(GTK_EDITABLE(GTK_COMBO(wTakk)^.entry),FALSE);
   gtk_widget_set_usize(wTakk,60,20);
   gtk_widget_set_usize(PGTKWIDGET(GTK_COMBO(wTakk)^.entry),60,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wTakk),345,35);
   g_s1:='1M'+#0+'3M'+#0+'10M'+#0;
   List1:=nil;
   List1:=g_list_append(List1,@g_s1[1]);
```

```
   List1:=g_list_append(List1,@g_s1[4]);
   List1:=g_list_append(List1,@g_s1[7]);
   gtk_combo_set_popdown_strings(GTK_COMBO(wTakk),List1);
   wText7:=gtk_entry_new; gtk_widget_set_usize(wText7,45,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText7),345,60);
   gtk_entry_set_max_length(PGTKENTRY(wText7),4);
   wText8:=gtk_entry_new; gtk_widget_set_usize(wText8,70,23);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wText8),495,10);

   gtk_entry_set_text(PGTKENTRY(wText1),'400');
   gtk_entry_set_text(PGTKENTRY(wText2),'410');
   gtk_entry_set_text(PGTKENTRY(wText3),'1.0');
   gtk_entry_set_text(PGTKENTRY(wText4),'Proovim66tmine');
   gtk_entry_set_text(PGTKENTRY(wText5),'1000');
   gtk_entry_set_text(PGTKENTRY(wText7),'100');
   gtk_entry_set_text(PGTKENTRY(wText8),'ajutine');

   wNupp2:=gtk_button_new_with_label('Ava...'); gtk_widget_set_usize(wNupp2,80,25);
   gtk_signal_connect(pGtkObject(wNupp2), 'clicked',GTK_SIGNAL_FUNC(@AvaProc),@g_MainWindow);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wNupp2),600,40);
   wNupp3:=gtk_button_new_with_label('Abi'); gtk_widget_set_usize(wNupp3,80,25);
   gtk_signal_connect(pGtkObject(wNupp3), 'clicked',GTK_SIGNAL_FUNC(@HelpProc),@g_MainWindow);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wNupp3),600,70);

   wNupp4:=gtk_button_new_with_label('<'); gtk_widget_set_usize(wNupp4,15,15);
   gtk_signal_connect(pGtkObject(wNupp4), 'clicked',
                      GTK_SIGNAL_FUNC(@AllaProc),{@g_MainWindow}nil);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wNupp4),650,110);
   wNupp5:=gtk_button_new_with_label('>'); gtk_widget_set_usize(wNupp5,15,15);
   gtk_signal_connect(pGtkObject(wNupp5), 'clicked',
                      GTK_SIGNAL_FUNC(@YlesProc),{@g_MainWindow}nil);
   gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wNupp5),665,110);

   with Nupp1
   do begin
      wNupp1:=gtk_button_new_with_label('Nupp1');
      gtk_widget_set_usize(wNupp1,80,25);
      gtk_signal_connect(pGtkObject(wNupp1),'clicked',
                         GTK_SIGNAL_FUNC(@StartStopProc),@g_MainWindow);
      gtk_fixed_put(PGTKFIXED(wBox0),pGtkWidget(wNupp1),600,10);
      glist1:=gtk_container_children(PGTKCONTAINER(wNupp1));
      label1:=PGTKLABEL(glist1^.data);
      gtk_label_set_text(PGTKLABEL(label1),'Start');
      end;  {with}
   end;  //with Result^
{}
gtk_timeout_add(1000,@KellProc,nil);
{}
FD:=fdOpen('/dev/ttyS0',Open_RdWr or Open_Sync);
if ((FD=-1) or (not FLock(FD,LOCK_EX)))
then begin
     writeln('port ei avane v6i ei lukustu');
     exit;
     end;
TCGetAttr(FD,tios);
tios.c_cflag:=CS8 or CREAD or CLOCAL;
tios.c_lflag:=0;
tios.c_oflag:=0;
tios.c_iflag:=0;
tios.c_cc[VMIN]:=0;
CFSetOSpeed(tios,B19200);
CFSetISpeed(tios,B19200);
TCSetAttr(FD,TCSANOW,tios);
IOCtl(FD,TIOCMBIC,@DTR);
{}
gtk_widget_show_all(PGtkWidget(g_MainWindow.Window));
gtk_main();
end.
```

Lisa II
Spektromeetriga DFS-52 mõõtekompleks

PIC-kontrolleri A2 programm

```
;*******************************************************
;                                                      **
;                    RAMAN1.ASM                        **
;                                                      **
;*******************************************************
; 19. aprillil 2001. a.

; RA0 - F3 - katiku andur
; RA1 - S1/2
; RA2 - S1/4
; RA3 - S1/8
; RA4 - S1/16
; RB0 - F4 - katiku ajur
; RB1 - CDA2
; RB2 - CDA1
; RB3 - CRG
; RB4 - DRG ja loenduri RESET
; RB5 - RS-232-siini valjund
; RB6 -      "       sisend
; RB7 - juhtsignaal PICII-le

;konstandid:
inbit           equ     6       ;RB6 on RS-232 siini sisend
outbit          equ     5       ;RB5  "    "     "   v,ljund
DRG             equ     4       ;
CRG             equ     3       ;
CDA1            equ     2       ;
CDA2            equ     1       ;
F3              equ     0       ;katiku andur
F4              equ     0       ;katiku ajur
RESET           equ     4       ;loenduri reset
pikkus          equ     0x1e    ;maarab baud rate
bytet           equ     0x21    ;saadetav bait
byter           equ     0x22    ;vastu v,etud bait
RxCount         equ     0x23    ;kasut. RS-232 juures loendurina
hi              equ     0x26    ;vanema baidi hoidmiseks kus vaja
lo              equ     0x24    ;signaali keskmine bait
noorim          equ     0x29    ;signaali noorimad bitid
loend1          equ     0x27    ;loendur aja jaoks
loend2          equ     0x28    ;      "
paus            equ     0x2a    ;

include         "pic84v4.h"

;*********************************************************************

        goto            Start
        nop
        nop
        nop

;*********************************************************************

katkestus
        incf            hi, f                   ;korgem bait
        bcf             INTCON, T0IF            ;lipp maha
        retfie                                  ;
; katkestuse lopp

;*********************************************************************

Start
        bcf             INTCON,GIE              ;keelata katkestused
        bsf             STATUS, RP0             ;panga valik
        movlw           0xff                    ;
```

```
        movwf           portA                   ;portA konf.
        movlw           0xe0                    ;
        movwf           portB                   ;portB konf.

        bsf             OPTION, RTE             ;frondi maaramine (langev)
        bsf             OPTION, PSA             ;prescaler kellale
        bsf             OPTION, RTS             ;counter mode
        bcf             STATUS, RP0             ;panga valik
        clrf            hi                      ;DA v@@rtus
        clrf            byter                   ;DA v@@rtus
        call            danullimine             ;
        bsf             portB, CDA1             ;
        bsf             portB, CDA2             ;
        nop
        nop
        nop
        bcf             portB, CDA1             ;
        bcf             portB, CDA2             ;

;*****************************************************************************
;
;       Kasud
;
;       L - lugeda andurid
;       D - andmed DA1-e
;       d - andmed DA2-e
;       A - lugeda pulsse

Commands
        btfsc           portB, 7
        goto            Commands
        call            Rx

        movlw           'L'
        xorwf           byter,w
        btfsc           status,Z
        goto            andurid

        movlw           'D'
        xorwf           byter,w
        btfsc           status,Z
        goto            da1

        movlw           'd'
        xorwf           byter,w
        btfsc           status,Z
        goto            da2

        movlw           'A'
        xorwf           byter,w
        btfsc           status,Z
        goto            pulsid

        nop                                     ;seni teine PIC
        nop                                     ;   jouab keelu
        nop                                     ;   peale panna,
        nop                                     ;   kui vaja
        nop
        nop
        nop
        nop
        nop
        nop
        goto            Commands                ; vale kask

;*****************************************************************************

; teatame, et valmis
OK
        bsf             STATUS, RP0
        bsf             portB, 7                ;RB7 sisendiks
        bcf             STATUS, RP0
        clrf            bytet
        call            Tx_Byte
        goto            Commands

;*****************************************************************************
```

```
pulsid
        bsf             portB, RESET            ;loendur nulli
        clrf            hi                      ;
        clrf            TMR0                    ;
        bsf             INTCON, T0IE            ;
        bcf             INTCON, T0IF            ;lipu puhastamine
        bsf             INTCON,GIE              ;katkestuste lubamine
        bcf             portB, RESET            ;loendur k@ima
tsykkel
        btfss           portB, 7                ;teiselt PIC-lt
        goto            tsykkel                 ;
        movf            portA, w                ;noorimad bitid
        bsf             portB, RESET            ;loendur seisma
        bcf             INTCON,GIE              ;keelame katkestused
        movwf           bytet                   ;noorim bait
        rrf             bytet, f                ;
        movlw           0x0f                    ;
        andwf           bytet, f                ;
        btfss           bytet, 3                ;kas nullimisel
        goto            passi                   ; toimus ylekanne?
        decf            TMR0, f                 ;
        incf            TMR0, w                 ;
        btfsc           STATUS, Z               ;kas toimus
        decf            hi, f                   ; ylekanne vanimas?
passi
        btfsc           portB, 7                ;teiselt PIC-lt
        goto            passi                   ;
        call            Tx_Byte                 ;noorim bait
        movf            TMR0, w                 ;keskmine bait
        movwf           bytet                   ;
        call            Tx_Byte                 ;
        movf            hi, w                   ;vanim bait
        movwf           bytet                   ;
        call            Tx_Byte                 ;
        goto            OK

;*********************************************************************

andurid
        call            kaja                    ;
        goto            OK                      ;done

;*****************************************************************
;   Pingete osa praktiliselt sama, ainult registrid muudetud
da1
        bsf             STATUS, RP0
        bcf             portB, 7                ;teisele PIC-le
        bcf             STATUS, RP0
        bsf             portB, 7                ;
        call            kaja                    ;
        call            dale                    ;loeb DA-sse v,,rtuse
        bsf             portB, CDA1             ;stroob DA1-le
        nop
        nop
        nop
        bcf             portB, CDA1             ;
        goto            OK
da2
        bsf             STATUS, RP0
        bcf             portB, 7                ;teisele PIC-le
        bcf             STATUS, RP0
        bsf             portB, 7                ;
        call            kaja                    ;
        call            dale                    ;sama DA2 jaoks
        bsf             portB, CDA2             ;
        nop
        nop
        nop
        bcf             portB, CDA2             ;
        goto            OK
dale
        call            Rx                      ; loeme korgema poole
        call            kaja                    ;
        movf            byter,0
        movwf           hi
        call            Rx                      ; loeme madalama poole
```

```
        call            kaja                    ;
danullimine
        movlw           8
        movwf           RxCount
dale1
        bcf             portB, DRG
        btfsc           byter,0
        bsf             portB, DRG              ; registrile andmed peale
        bsf             portB, CRG              ; registrile clock peale
        nop
        nop
        nop
        bcf             portB, CRG              ; registrilt clock maha
        rrf             byter,1                 ;nihutab uue biti ette
        decfsz          RxCount,1               ;loendab tsykleid
        goto            dale1                   ;kokku 8 korda
        movlw           4
        movwf           RxCount
dale2                                           ;sama, mis dale1,
        bcf             portB, DRG              ;   ainult 4 korda
        btfsc           hi, 0
        bsf             portB, DRG              ; registrile andmed peale
        bsf             portB, CRG              ; registrile clock peale
        nop
        nop
        nop
        bcf             portB, CRG              ; registrilt clock maha
        rrf             hi, 1
        decfsz          RxCount,1
        goto            dale2
        return                                  ; nihkeregistri lopp

;*********************************************************************

; RS232 jupid
delay
        movwf           paus
delay1
        decfsz          paus,f  ;1   pausi pikkus = W * 3 + 3 = (W+1)*3 us
        goto            delay1                  ;2
        return                                  ;2

;*********************************************************************

Rx
        btfss           portB,inbit             ;ootame t□hja liini
        goto            Rx
Rx_aa2
        btfsc           portB,inbit             ;ootame startbitti
        goto            Rx_aa2
        movlw           0xe                     ;paus tuleb 1+2+2+W*3=53
        call            delay
;Rx_aa3
         btfsc          portB,inbit             ;akki on startbitt kadunud 54
         goto           Rx                      ;ootame otsast peale      55
         movlw          8                       ;                          56
         movwf          RxCount                 ;                          57
bittaa2
        movlw           pikkus                          ;1 58 61
        call            delay                           ;4+W*3 57
        bcf             status,C                ;1 54
        btfsc           portB,inbit             ;1 55
        bsf             status,C                ;1 56
        rrf             byter,f                 ;1 57
        decfsz          RxCount,f               ;1 58
        goto            bittaa2                 ;2 60
        return

;*********************************************************************

Tx_Byte
        bsf             portB, outbit           ;
        bsf             STATUS, RP0             ;panga valik
        bcf             portB, outbit           ;outbit valjundiks
        bcf             STATUS, RP0             ;panga valik
        movlw           pikkus                  ;eelmise baidi stopp-bitt
        call            delay                   ;   kestab =>94  us
```

```
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        bcf             portB, outbit           ;startbitt
        movlw           pikkus                  ;1
        call            delay                   ;2
        movlw           8                       ;1
        movwf           RxCount                 ;1
tx
        nop
        nop
        nop
        btfss           bytet,0                 ;1
        goto            null                    ;2
        bsf             portB, outbit           ;1
        goto            xxx                     ;2
null
        bcf             portB, outbit           ;1
xxx
        rrf             bytet,f                 ;1
        movlw           pikkus                  ;1
        call            delay                   ;2
        decfsz          RxCount,f               ;1
        goto            tx                      ;2
        bsf             portB, outbit           ;stoppbitt
        rrf             bytet,f
        bsf             STATUS, RP0             ;panga valik
        bsf             portB, outbit           ;outbit sisendiks
        bcf             STATUS, RP0             ;panga valik
        return

;*********************************************************************

kaja
        movf            byter, w                ;
        movwf           bytet                   ;
        call            Tx_Byte                 ;
        return

;*********************************************************************

        org             0x2007                  ;konf. fuses
        data            0x19                    ;XT-osc, CP off ja PWRTE
        end
```

PIC-kontrolleri A3 programm

```
;*******************************************************
;                                                      **
;                   RAMAN2.ASM                         **
;                                                      **
;*******************************************************
; 19. aprillil 2001. a.

; RA0 - p""re
; RA1 - läpp
; RA2 - F1 - filtri andur
; RA3 - vaba
; RA4 - S2/16
; RB0 - faas1
; RB1 - faas2
; RB2 - faas3
; RB3 - faas4
; RB4 - F2 - filtri ajur
; RB5 - RS-232 v,ljund
; RB6 - RS-232 sisend
; RB7 - esimeselt PIC-lt

;konstandid:
inbit           equ     6       ;RB6 on RS-232 siini sisend
outbit          equ     5       ;RB5  "     "      "   v,ljund
F1              equ     2       ;filtri andur
F2              equ     4       ;filtri ajur
pikkus          equ     0x1e    ;maarab baud rate
pclath          equ     0x0a
loend2          equ     0x14    ;kasutatakse sekundi formeerimisel
mootor          equ     0x15    ;alumine pool on mahiste mask
okay            equ     0x16    ;
abi             equ     0x17    ;siin kujundatakse portB
loend1          equ     0x18    ;m66tmise aeg
time            equ     0x20    ; faasilt faasile aeg
bytet           equ     0x21    ;saadetav bait
byter           equ     0x22    ;vastu v,etud bait
RxCount         equ     0x23    ;kasut. DA ja RS-232 juures loendurina
hi              equ     0x26    ;vanema baidi hoidmiseks kus vaja
paus            equ     0x2a    ;kasut. viivituse juures

include         "pic84v4.h"

;*********************************************************************

        goto            Start
        nop
        nop
        nop

;*********************************************************************

katkestus
        bcf             INTCON, T0IF            ;lipp maha
        incf            hi, f                   ;
        retfie                                  ;
; katkestuse lopp

;*********************************************************************

Start
        bcf             INTCON,GIE              ;keelata katkestused
        bsf             STATUS, RP0             ;panga valik
        movlw           0xff                    ;portA konf.
        movwf           portA                   ;
        movlw           0xe0                    ;pordi B konfig.: sssvvvvv
        movwf           portB                   ;

        bcf             OPTION, RTE             ; frondi maaramie (kasvav)
        bsf             OPTION, PSA             ; prescaler kellale
        bsf             OPTION, RTS             ; counter mode
        bcf             STATUS, RP0             ;panga valik
        clrf            portB
```

```
        clrf            okay                    ;
        clrf            mootor                  ;

;*****************************************************************************
;
;       Kasud
;
;       M - mootor edasi n sammu
;       m - mootor tagasi n sammu
;       I - filter ette
;       P - filter „ra
;       A - lugeda pulsse const. aja jooksul
;       S - sammumootor starti
;       K - m66tmise aeg

Commands
        btfsc           portB, 7                ;
        goto            Commands                ;
        call            Rx

        movlw           'K'
        xorwf           byter, w
        btfsc           status, Z
        goto            kestus

        movlw           'f'
        xorwf           byter, w
        btfsc           status, Z
        goto            faas_maha

        movlw           'S'
        xorwf           byter,w
        btfsc           status,Z
        goto            stardi_lyliti

        movlw           'M'
        xorwf           byter,w
        btfsc           status,Z
        goto            sammumootor1

        movlw           'm'
        xorwf           byter,w
        btfsc           status,Z
        goto            sammumootor2

        movlw           'I'
        xorwf           byter,w
        btfsc           status,Z
        goto            filter_ette

        movlw           'P'
        xorwf           byter,w
        btfsc           status,Z
        goto            filter_ara

        movlw           'A'
        xorwf           byter,w
        btfsc           status,Z
        goto            pulsid

        nop                                     ;selle ajaga jouab
        nop                                     ;   teine PIC keelu
        nop                                     ;   yles panna,
        nop                                     ;   kui vaja
        nop
        nop
        nop
        nop
        nop
        nop
        goto            Commands                ; vale kask

;*****************************************************************************

; teatame, et valmis
OK
        bsf             STATUS, RP0             ;panga valik
```

```
        bsf             portB, 7                ;tegelt trisB
        bcf             STATUS, RP0             ;panga valik
        movf            okay, w
        movwf           bytet
        clrf            okay
        call            Tx_Byte
        goto            Commands

;*********************************************************************

kestus
        bsf             STATUS, RP0             ;
        bcf             portB, 7                ;RB7 v@ljundiks
        bcf             STATUS, RP0             ;
        bsf             portB, 7                ;
        call            kaja                    ;
        call            Rx                      ;argument
        call            kaja                    ;
        movf            byter, w                ;
        movwf           loend1                  ;
        goto            OK                      ;


;*********************************************************************

stardi_lyliti
        call            kaja                    ;
        movlw           0xc0
        movwf           time
        movlw           10                      ;
        movwf           byter                   ;
start1
        call            Faas_peale

; kas lopp on kaes?
        btfss           portA, 1
        goto            mark000
        incf            mootor,f
        movlw           7
        andwf           mootor,f
        goto            start1
mark000
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        call            sekund
        movlw           0xc0
        movwf           time
start2
; kas lopp on kaes?
        btfsc           portA, 1
        goto            mar00
        decf            mootor, f
        movlw           7
        andwf           mootor,f
        call            Faas_peale
        goto            start2
mar00
        btfss           portA, 0                ;poore
        goto            OK                      ;kui must -> valge
        decf            mootor, f               ;
```

```
        movlw           7                       ;
        andwf           mootor, f               ;
        call            Faas_peale              ;
        goto            mar00                   ;

;*********************************************************************

pulsid
        movf            loend1, w               ;
        movwf           RxCount                 ;
        clrf            hi                      ;
        clrf            TMR0                    ;
        bcf             INTCON, T0IF            ;
        bsf             INTCON, T0IE            ;katkestuste lubamine
        bsf             INTCON, GIE             ;
tsykkel
        call            sekund                  ;
        call            sekund                  ;
        call            sekund                  ;
        call            sekund                  ;
        decfsz          RxCount, f              ;
        goto            tsykkel                 ;
        bcf             INTCON, GIE             ;katkestuste keelamine
        movf            TMR0, w                 ;noorem bait
        bsf             STATUS, RP0             ;
        bcf             portB, 7                ;RB7 v@ljundiks
        bcf             STATUS, RP0             ;
        bsf             portB, 7                ;teisele PIC-le
        movwf           bytet                   ;noorem bait
        call            TxByte                  ;
        movf            hi, w                   ;vanem bait
        movwf           bytet                   ;
        call            Tx_Byte                 ;
        bsf             STATUS, RP0             ;
        bsf             portB, 7                ;teisele PIC-le
        bcf             STATUS, RP0             ;
        goto            Commands

;*********************************************************************

sekund                                          ;sama, mis enne
        clrf            loend2
sek
        nop
        nop
        nop
        nop
        nop
        nop
        decfsz          loend2, f
        goto            sek
        return

;*********************************************************************

filter_ette
        call            kaja                    ;
        bsf             portB, 4                ;RB4 on filtri ajur
        btfsc           portA, 2                ;filtri andur
        incf            okay, f                 ;
        goto            OK                      ;done
filter_ara
        call            kaja
        bcf             portB, 4
        goto            OK

;*********************************************************************

; Sammumootori ajamine lopulyliti poole

sammumootor1                                    ;ajab mootorit tagasi
                                                ;   byter+256*hi sammu
        bsf             STATUS, RP0             ;panga valik
        bcf             portB, 7                ;teisele PIC-le
        bcf             STATUS, RP0             ;panga valik
        bsf             portB, 7                ;
        call            kaja                    ;
```

```
        call            Rx                      ;loeme argumendid:
        call            kaja                    ;
        movf            byter,0                 ;   vanem bait on hi-s,
        movwf           hi                      ;
        call            Rx                      ;   noorem byter-s
        call            kaja                    ;
        movlw           0xc0                    ;
        movwf           time                    ;
algus1
        bcf             STATUS, Z               ;
        movf            byter, f                ;kas madalam on 0?
        btfss           STATUS, Z               ;
        goto            pole_01                 ;kui pole, siis ...
        bcf             STATUS, Z               ;
        movf            hi, f                   ;kui on, siis kontrollime
        btfsc           STATUS, Z               ;   kärgemat
        goto            OK                      ;kui see ka 0, siis l,pp
        decf            hi, f                   ;kui pole, siis mälemas dec
pole_01
        decf            byter, f                        ;muidu ainult nooremas

        incf            mootor, f
        movlw           7                       ;
        andwf           mootor,f                ;
        call            Faas_peale

        goto            algus1                  ;uuele ringile

;*****************************************************************************

; Sammumootori ajamine alguse poole (skaneerimine)
sammumootor2

        bsf             STATUS, RP0             ;panga valik
        bcf             portB, 7                ;
        bcf             STATUS, RP0             ;panga valik
        bsf             portB, 7                ;

        call            kaja                    ;
        call            Rx                      ;loeme argumendid
        call            kaja                    ;
        movf            byter,0                 ;
        movwf           hi                      ;
        call            Rx                      ;
        call            kaja                    ;

        movlw           0xc0                    ;
        movwf           time                    ;
algus2
        bcf             STATUS, Z               ;
        movf            byter, f                ;Käik sama, mis eelmisel,
        btfss           STATUS, Z               ;   ainult incf mootor
        goto            pole_02                 ;   asemel on decf mootor
        bcf             STATUS, Z               ;
        movf            hi, f                   ;
        btfsc           STATUS, Z               ;
        goto            OK                      ;
        decf            hi, f                   ;
pole_02
        decf            byter, f                ;
        decf            mootor,f                ;
        movlw           7                       ;
        andwf           mootor,f                ;
        call            Faas_peale              ;

        goto            algus2                  ;

;*****************************************************************

Faas_peale
        movlw           0xf0                    ;
        andwf           portB, w                ;
        movwf           abi                     ;

        call            Faas
        iorwf           abi, w
        movwf           portB
```

```
        call            dectime                 ;praegu ei deci
        movf            time,0
        call            viivitus

        return

;*****************************************************************

faas_maha
        call            kaja
no_phase
        movlw           0xf0
        andwf           portB, f

        goto            OK

;*****************************************************************

dectime
;       btfsc           time,7
;       decf            time,f
        return

;*****************************************************************

viivitus
        movwf           paus
viivitus1
        decfsz          paus,f
        goto            viivitus1
        return

;*****************************************************************

; RS232 jupid

delay
        movwf           paus
delay1
        decfsz          paus,f  ;1   pausi pikkus = W * 3 + 3 = (W+1)*3 us
        goto            delay1                  ;2
        return                                  ;2

;*****************************************************************

Rx
        btfss           portB,inbit             ;ootame t□hja liini
        goto            Rx

Rx_aa2
        btfsc           portB,inbit             ;ootame startbitti
        goto            Rx_aa2
        movlw           0xe                     ;paus tuleb 1+2+2+W*3=53
        call            delay

;Rx_aa3
         btfsc          portB,inbit             ;akki on startbitt kadunud 54
         goto           Rx                      ;ootame otsast peale      55
         movlw          8                       ;                          56
         movwf          RxCount                 ;                          57
;
bittaa2
        movlw           pikkus                          ;1 58 61
        call            delay                           ;4+W*3 57

        bcf             status,C                ;1 54
        btfsc           portB,inbit             ;1 55
        bsf             status,C                ;1 56

        rrf             byter,f                 ;1 57
        decfsz          RxCount,f               ;1 58
        goto            bittaa2                 ;2 60
        return

;*****************************************************************
```

```
Tx_Byte
        bsf             STATUS, RP0             ;panga valik
        bcf             portB, outbit           ;outbit valjundiks
        bcf             STATUS, RP0             ;panga valik
        bsf             portB, outbit           ;

        movlw           pikkus                  ;eelmise baidi stopp-bitt
        call            delay                   ;
        nop
        nop
        nop
        nop
        nop
        bcf             portB, outbit
        movlw           pikkus                  ;1
        call            delay                   ;2

        movlw           8                       ;1
        movwf           RxCount                 ;1
tx
        nop
        nop
        nop
        btfss           bytet,0                 ;1
        goto            null                    ;2
        bsf             portB, outbit           ;1
        goto            xxx                     ;2
null
        bcf             portB, outbit           ;1
xxx
        rrf             bytet,f                 ;1
        movlw           pikkus                  ;1
        call            delay                   ;2
        decfsz          RxCount,f               ;1
        goto            tx                      ;2
        bsf             portB, outbit           ;stoppbitt
        rrf             bytet,f

        bsf             STATUS, RP0             ;panga valik
        bsf             portB, outbit           ;outbit sisendiks
        bcf             STATUS, RP0             ;panga valik

        return

;*****************************************************************

kaja
        movf            byter,0
        movwf           bytet
        call            Tx_Byte
        return

;*****************************************************************************

        org             0x300

Faas
        movlw           0x03
        movwf           pclath
        movf            mootor,w
        addwf           pcl,f

        retlw           0x0c
        retlw           0x08
        retlw           0x09
        retlw           0x01
        retlw           0x03
        retlw           0x02
        retlw           0x06
        retlw           0x04

        org             0x2007                  ;configuration fuses
        data            0x19                    ;XT-osc, CP off ja PWRTE

        end
```

Windows-keskonna juhtprogramm

```
program R95_run;

{fp:}
{$APPTYPE GUI}
uses Windows, SysUtils;

{Delphi:}
(*uses Windows, Messages, SysUtils, CommDlg;*)

type andmed = record
              algus, l6pp, aeg, myra1, myra2, laser1, laser2: longint;
              samm, pinge1, pinge2, punkte, max1, max2, xnr, x0, dx3, dx4, nih: longint;
              k1, dx1, dx2, dx5, k1min, k1max: real;
              komm: array[0..255] of char;
              kuup: array[0..20] of char;
              nimi, laiend: array[0..50] of char;
              end;
     pAndmed = ^andmed;

var g_hwMain: HWND;
    g_DataPath: array[0..255] of char;
    g_x, g_l1: longint;
    g_Reeper1, g_Reeper2: longint;
    g_i1, g_Punkt: integer;
    g_Font1: HFONT;
    g_HallPintsel: HBRUSH;
    g_m1, g_m2: array[0..5000] of longint;
    g_AvaHeader, g_DefaultHeader, g_T66Header: andmed;
    g_Filter, g_Init, g_M66danMyra1: LONGBOOL;
    g_bl: LONGBOOL;
    g_MustSulg, g_HallSulg, g_PunaneSulg: HPEN;
    g_TRGraafik, g_TRxtelg: TRECT;
    g_HRGraafik: HRGN;
    g_hCOM: tHandle;
    dw1, g_Viivitus: DWORD;
    g_Sodi: record yks: LONGBOOL; nr: longint; end;
    g_htr1: tHandle;

const g_Katkestan: LONGBOOL = FALSE;
      g_M66tmiseTyyp: byte = 0;

{$R R95RUN.RES}

procedure piip; begin messagebeep($FFFF); end;

procedure Status(s: string);
begin
s:=s+#0;
SetDlgItemText(g_hwMain,122,@s[1]);
end;  {status1}

procedure Status2s(s: string);
begin
s:=s+#0;
SetDlgItemText(g_hwMain,123,@s[1]);
end;

function TextToFloat(p: pChar; l: integer; var r: real): LONGBOOL;
var i, j, k: integer;
    c: char;
    r1, r2: extended;
    buf1, buf2, buf3: array[0..20] of char;
label 1, 2, 3;
begin
TextToFloat:=FALSE;
Move(p^,buf1,20);
i:=0;
j:=0;
k:=0;
1:
c:=buf1[i];
if i>10 then exit;  {jama}
```

```
if (c in ['0'..'9']) then begin buf2[j]:=c; inc(j); inc(i); goto 1; end;
if c='.' then begin buf2[j]:=#0; inc(i); goto 2; end;
if c=#0 then begin buf2[j]:=#0; buf3[0]:=#0; goto 3; end;
exit;  {jama}
2:
c:=buf1[i];
if i>10 then exit;  {jama}
if (c in ['0'..'9']) then begin buf3[k]:=c; inc(k); inc(i); goto 2; end;
if c=#0 then begin buf3[k]:=#0; goto 3; end;
exit;  {jama}
3:
r1:=StrToIntDef(StrPas(@buf2),0);
r2:=StrToIntDef(StrPas(@buf3),0);
for i:=1 to k do r2:=r2/10;
r:=r1+r2;
TextToFloat:=TRUE;
end;

function PordiViivitus(dw: DWORD): LONGBOOL;
var CTO: TCOMMTIMEOUTS;
begin
PordiViivitus:=FALSE;
with CTO do begin
            ReadIntervalTimeout:=0;
            ReadTotalTimeoutMultiplier:=dw;
            ReadTotalTimeoutConstant:=0;
            WriteTotalTimeoutMultiplier:=0;
            WriteTotalTimeoutConstant:=0;
            end;
if not SetCommTimeouts(g_hCOM,CTO) then exit;
PordiViivitus:=TRUE;
end;

function AvaPort: LONGBOOL;
var dcb: TDCB;
begin
AvaPort:=FALSE;
g_hCOM:=CreateFile('COM1',GENERIC_READ+GENERIC_WRITE,0,nil,OPEN_EXISTING,
                   FILE_ATTRIBUTE_NORMAL,0);
if g_hCOM=INVALID_HANDLE_VALUE then exit;
if not SetupComm(g_hCOM,16,16) then exit;
if not GetCommState(g_hCOM,dcb) then exit;
with dcb do begin BaudRate:=9600; ByteSize:=8; Parity:=0; StopBits:=0; end;  {with}
if not SetCommState(g_hCOM,dcb) then exit;
g_Viivitus:=1000;
if not PordiViivitus(g_Viivitus) then exit;
AvaPort:=TRUE;
end;

function ResetPort: LONGBOOL;
begin
ResetPort:=FALSE;
EscapeCommFunction(g_hCOM,SETDTR);  {reset PIC-le}
sleep(100);
EscapeCommFunction(g_hCOM,CLRDTR);
sleep(100);
if not PurgeComm(g_hCOM,PURGE_TXABORT+PURGE_RXABORT+PURGE_TXCLEAR+PURGE_RXCLEAR) then exit;
ResetPort:=TRUE;
end;

function SulePort: LONGBOOL;
begin
if CloseHandle(g_hCOM) then SulePort:=TRUE else SulePort:=FALSE;
end;

function T66Proc0(pl: plongint): DWORD; stdcall;  {Init}
var b: byte;
    hw: HWND;  {aken, kuhu teated saata}
    dw: DWORD;
    bl: LONGBOOL;
    buf_out: array[1..10] of char;
label 1, 3;
begin
T66Proc0:=0;
hw:=pl^;
if not PordiViivitus(3000) then goto 1;
buf_out[1]:='p';  {katik kinni}
```

```
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
buf_out[1]:='P';  {filter eest}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
g_Filter:=FALSE;
buf_out[1]:='D';  {I kanali k6rgepinge}
dw:=round(1.86*g_T66Header.pinge1);
buf_out[2]:=char(dw div 256);  {argumendid vahetada  !!!!!}
buf_out[3]:=char(dw mod 256);  {!!!!!}
bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
buf_out[1]:='d';  {II kanali k6rgepinge}
dw:=round(1.86*g_T66Header.pinge2);
buf_out[2]:=char(dw div 256);  {!!!!!}
buf_out[3]:=char(dw mod 256);  {!!!!!}
bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if not PordiViivitus(240000) then goto 1;  {?????}
buf_out[1]:='S';  {alguse otsimine}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if g_Katkestan then goto 3;
g_x:=g_Reeper1;
PostMessage(hw,WM_COMMAND,501,0);  {k6ik korras}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,501,1);  {PIC ei vasta}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,501,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc0}

function T66Proc1(pl: plongint): DWORD; stdcall;  {m66tmine ja sammumine}
var punkt: integer;
    b: byte;
    hw: HWND;
    dw: DWORD;
    l, l1, l2, l3: longint;
    bl: LONGBOOL;
    buf_in, buf_out: array[1..10] of char;
label 1, 3;
begin
T66Proc1:=0;
hw:=pl^;
{g_Viivitus:=1000+round(1.6*g_T66Header.samm);}
g_Viivitus:=3000;
if not PordiViivitus(g_Viivitus) then goto 1;
g_Punkt:=0;
punkt:=0;
l:=0;
buf_out[1]:='A';  {loendamine}
buf_out[2]:='m';  {n sammu edasi}
buf_out[3]:=char(g_T66Header.samm div 256);
buf_out[4]:=char(g_T66Header.samm mod 256);
buf_out[5]:='i';  {katik lahti}
buf_out[6]:='p';  {katik ette}
buf_out[7]:='I';  {filter ette}
buf_out[8]:='P';  {filter @ra}
with g_T66Header do if ((algus>=laser1) and (algus<=laser2))
                    then begin  {filter ette}
                         bl:=WriteFile(g_hCOM,buf_out[7],1,dw,nil);
                         if ((not bl) or (dw<>1)) then goto 1;
                         bl:=ReadFile(g_hCOM,b,1,dw,nil);
                         if ((not bl) or (dw<>1)) then goto 1;
                         g_Filter:=TRUE;
                         end;
if g_Katkestan then goto 3;
bl:=WriteFile(g_hCOM,buf_out[5],1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {katik lahti}
{-------------------------------------------------------------------------}
if g_Katkestan then goto 3;  {I punkt}
g_Viivitus:=1000+round(1.6*g_T66Header.samm);
```

```
if not PordiViivitus(g_Viivitus) then goto 1;
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,buf_in,6,dw,nil); if ((not bl) or (dw<>6)) then goto 1;
l1:=ord(buf_in[1]);
l2:=ord(buf_in[2]);
g_m2[0]:=l1+256*l2;  {v6rdluskanal}
l1:=ord(buf_in[3]);
l2:=ord(buf_in[4]);
l3:=ord(buf_in[5]);
g_m1[0]:=l1+16*l2+16*256*l3;  {otsekanal}
if l<>0 then g_T66Header.k1:=180/l else g_T66Header.k1:=1;  {?????}
g_T66Header.max1:=0;
g_T66Header.max2:=0;
{PostMessage(hw,WM_COMMAND,502,0);}
repeat
if g_Katkestan then goto 3;
with g_T66Header do if (not g_Filter
                    and (g_x<laser1) and ((g_x+samm)>=laser1))
                  then begin  {filter ette}
                       if not PordiViivitus(3000) then goto 1;
                       bl:=WriteFile(g_hCOM,buf_out[7],1,dw,nil);
                       if ((not bl) or (dw<>1)) then goto 1;
                       bl:=ReadFile(g_hCOM,b,1,dw,nil);
                       if ((not bl) or (dw<>1)) then goto 1;
                       g_Filter:=TRUE;
                       if not PordiViivitus(g_Viivitus) then goto 1;
                       end;
bl:=WriteFile(g_hCOM,buf_out[2],3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
inc(g_x,g_T66Header.samm);
if g_Katkestan then goto 3;
with g_T66Header do if (g_Filter
                    and (g_x>laser2) and ((g_x-samm)<=laser2))
                  then begin  {filter eest}
                       if not PordiViivitus(3000) then goto 1;
                       bl:=WriteFile(g_hCOM,buf_out[8],1,dw,nil);
                       if ((not bl) or (dw<>1)) then goto 1;
                       bl:=ReadFile(g_hCOM,b,1,dw,nil);
                       if ((not bl) or (dw<>1)) then goto 1;
                       g_Filter:=FALSE;
                       if not PordiViivitus(g_Viivitus) then goto 1;
                       end;
inc(punkt);
if ((punkt>560) and (g_M66tmiseTyyp=2)) then punkt:=0;
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,buf_in,6,dw,nil); if ((not bl) or (dw<>6)) then goto 1;
l1:=ord(buf_in[1]);
l2:=ord(buf_in[2]);
g_m2[0]:=l1+256*l2;  {v6rdluskanal}
l1:=ord(buf_in[3]);
l2:=ord(buf_in[4]);
l3:=ord(buf_in[5]);
g_m1[punkt]:=l1+16*l2+16*256*l3;  {otsekanal}
PostMessage(hw,WM_COMMAND,502,0);  {yks punkt m66detud}
until punkt=g_T66Header.punkte-1;
{-------------------------------------------------------------------------------}
(*
if (fl=true) and (x>joon3)
                     then begin
                          if not filter_ara then goto 45;
                          end;
                     if (fl=false) and (x<=joon1) and ((x+samme)>joon1)
                     then begin
                          if not filter_ette then goto 45;
                          end;
*)
g_Viivitus:=3000;
if not PordiViivitus(g_Viivitus) then goto 1;  {?????}
bl:=WriteFile(g_hCOM,buf_out[6],1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
if g_Filter
then begin  {filter eest}
     bl:=WriteFile(g_hCOM,buf_out[8],1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
     bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
     g_Filter:=FALSE;
     end;
g_Viivitus:=1000;
```

```
if not PordiViivitus(g_Viivitus) then goto 1;  {?????}
PostMessage(hw,WM_COMMAND,503,0);  {valmis}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {pordi jama}  {!!!!!}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc1}

function T66Proc2(pl: plongint): DWORD; stdcall;  {myra m66tmine}
var i: integer;
    l, l1, l2, l3: longint;
    hw: HWND;
    dw: DWORD;
    bl: LONGBOOL;
    buf_in, buf_out: array[1..10] of char;
label 1, 3;
begin
T66Proc2:=0;
hw:=pl^;
buf_out[4]:='A';
if not PordiViivitus(3000) then goto 1;
{}
for i:=1 to 10
do begin
   if g_Katkestan then goto 3;
   bl:=WriteFile(g_hCOM,buf_out[4],1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
   bl:=ReadFile(g_hCOM,buf_in,6,dw,nil); if ((not bl) or (dw<>6)) then goto 1;  {OK}
   l1:=ord(buf_in[3]);
   l2:=ord(buf_in[4]);
   l3:=ord(buf_in[5]);
   l:=l+l1+16*l2+16*256*l3;
   PostMessage(hw,WM_COMMAND,507,i*$10000);  {1 punkt m66detud}
   end;
{}
if g_M66danMyra1 then begin g_T66Header.myra1:=round(l/10); l:=1; end
                 else begin g_T66Header.myra2:=round(l/10); l:=2; end;
PostMessage(hw,WM_COMMAND,507,l);  {myra1 v. myra2 m66detud}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {PIC ei vasta}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc2}

function T66Proc3(pl: plongint): DWORD; stdcall;  {l6petamine}
var b: byte;
    hw: HWND;
    dw, dw1: DWORD;
    bl: LONGBOOL;
    buf_out: array[1..10] of char;
label 1;
begin
T66Proc3:=0;
hw:=pl^;
if not PordiViivitus(3000) then goto 1;
buf_out[1]:='p';  {katik ette}
buf_out[2]:='P';  {filter @ra}
buf_out[3]:='f';  {faas maha}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,dw1,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
bl:=WriteFile(g_hCOM,buf_out[2],1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
bl:=WriteFile(g_hCOM,buf_out[3],1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
PostMessage(hw,WM_COMMAND,504,0);  {k6ik korras}
ExitThread(0);
```

```
exit;
1:
PostMessage(hw,WM_COMMAND,509,1);  {pordi jama}  {!!!!!}
ExitThread(0);
end;  {T66Proc3}

function T66Proc4(pl: plongint): DWORD; stdcall;  {liikumine koordinaadile}
var i: integer;
    b: byte;
    l: longint;
    hw: HWND;
    dw: DWORD;
    bl: LONGBOOL;
    buf_out: array[1..10] of char;
label 1, 3;
begin
T66Proc4:=0;
hw:=pl^;
if not PordiViivitus(3000) then goto 1;
(*buf_out[1]:='p';  {katik kinni}
bl:=WriteFile(g_hCOM,buf_out,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}*)
buf_out[2]:='K';  {loendusaeg}
buf_out[3]:=char(g_T66Header.aeg);
bl:=WriteFile(g_hCOM,buf_out[2],2,dw,nil); if ((not bl) or (dw<>2)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
l:=g_T66Header.algus-g_x;
g_Viivitus:=1000+round(1.6*abs(l));
if not PordiViivitus(g_Viivitus) then goto 1;  {!!!!!}
{-----------------------}  {liikumine koordinaadile}
if l>0 then buf_out[1]:='m' else buf_out[1]:='M';
buf_out[2]:=#255;
buf_out[3]:=#255;
for i:=1 to (abs(l) div 65535)  {65536 !!!!!}
do begin
   if g_Katkestan then goto 3;
   bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
   bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
   if (l<0) then g_x:=g_x-65535 else g_x:=g_x+65535;
   if g_Katkestan then goto 3;
   sleep(1000);
   end;
l:=abs(g_T66Header.algus-g_x);
buf_out[2]:=char(l div 256);
buf_out[3]:=char(l mod 256);
bl:=WriteFile(g_hCOM,buf_out,3,dw,nil); if ((not bl) or (dw<>3)) then goto 1;
bl:=ReadFile(g_hCOM,b,1,dw,nil); if ((not bl) or (dw<>1)) then goto 1;  {OK}
g_x:=g_T66Header.algus;
{-----------------------}
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,506,0);  {k6ik korras}
ExitThread(0);
exit;
1:
if g_Katkestan then goto 3;
PostMessage(hw,WM_COMMAND,509,1);  {pordi jama}  {!!!!!}
ExitThread(0);
exit;
3:
PostMessage(hw,WM_COMMAND,509,3);  {kunstlik l6petamine}
ExitThread(0);
end;  {T66Proc4}

function OnNimi(p: pChar): LONGBOOL;  {?????}
var i, j: integer;
    buf: array[0..50] of char;
begin
StrLCopy(buf,p,50);
OnNimi:=FALSE;
i:=StrLen(buf);
if i=0 then exit;
{setwindowtext(g_hwmain,@s[1]);}
for j:=1 to i-1 do if (not(buf[j] in ['A'..'Z','a'..'z','0'..'9','_','-',' '])) then exit;
OnNimi:=TRUE;
end;

function L6iguReal(r: real; b: byte): string;
```

```
var l: longint;
    s: string;
begin
l:=trunc(r);
if l=r then str(l,s) else str(r:1:b,s);
L6iguReal:=s+#0;
end;

function StrToLong(var s: string; var l: longint): LONGBOOL;  {leiab stringist t@isarvu}
var i, j: integer;
begin
StrToLong:=FALSE;
i:=1;
while (not (s[i] in ['0'..'9'])) do begin s[i]:=#32; inc(i); end;
j:=i;
while (s[i] in ['0'..'9']) do inc(i);
s:=copy(s,j,i-j);
val(s,l,i);
if i=0 then StrToLong:=TRUE;
end;

function OpenFileName(b: byte; p: pointer): LONGBOOL;
var f: TOpenFileName;
    buf1, buf2: array[0..255] of char;
begin
OpenFileName:=FALSE;
FillChar(f,sizeOf(f),#0);
FillChar(buf1,SizeOf(buf1),#0);
FillChar(buf2,SizeOf(buf2),#0);
buf1:='dat file'+#0+'*.dat'+#0+'Any File'+#0+'*.*'+#0+#0;
f.lStructSize:=sizeof(f);
f.lpstrFilter:=@buf1;
f.nMaxFile:=sizeof(buf2);
f.lpstrFile:=@buf2;
f.lpstrInitialDir:=@g_DataPath;
{fp:}
if b=1 then OpenFileName:=GetOpenFileName(@f) else OpenFileName:=GetSaveFileName(@f);
{Delphi:}
(*if b=1 then OpenFileName:=GetOpenFileName(f) else OpenFileName:=GetSaveFileName(f);*)
StrLCopy(p,@buf2,255);
end;

function Salvesta(nimi: pChar): LONGBOOL;
var f: text;
    i: integer;
    r: real;
begin
Salvesta:=FALSE;
Assign(f,StrPas(nimi));
{$I-} ReWrite(f);  {$I+}
if IOResult<>0 then exit;
with g_T66Header
do begin
   writeln(f,'WinRaman andmefail');
   writeln(f,'02.02.2002 16:18');  {!!!!!!}
   writeln(f,'Algus: ',algus div 48,' cm-1');  {algus}
   writeln(f,'Samm: ',samm,' /48 cm-1');  {samm}
   writeln(f,'Katsepunkte: ',punkte);  {punkte}
   writeln(f,'Loendusaeg: ',10*aeg,' ms');  {aeg}  {!!!!!!}
   writeln(f,'I FEK anoodpinge: ',pinge1,'V');  {pinge1}
   writeln(f,'II FEK anoodpinge: ',pinge2,'V');  {pinge2}
   writeln(f,'Pimeimpulsse alguses: ',myra1);  {myra1}
   writeln(f,'Pimeimpulsse l6pus: ',myra2);  {myra2}
   writeln(f,'Rayleigh'' joone algus: ',laser1 div 48,' cm-1');  {laser1}
   writeln(f,'Rayleigh'' joone l6pp: ',laser2 div 48,' cm-1');  {laser2}
   writeln(f,'Kommentaar: ',komm);  {komm}
   for i:=1 to 3 do writeln(f);
   for i:=0 to punkte-1 do begin
                           r:=(algus div 48)+i*samm/48;
                           writeln(f,r:2:2,' ',g_m1[i],' ',g_m2[i]);
                           end;
   end;  {with}
Close(f);
end;  {salvesta}

function LoeSisse(nimi: pChar; var viga: integer): LONGBOOL;
var f: text;
```

```
    i: integer;
    r: real;
    s: string;
    bl: LONGBOOL;
label 1;
begin
LoeSisse:=FALSE;
Assign(f,StrPas(nimi));
IOResult;
{$I-} Reset(f);
if IOResult<>0 then begin viga:=-1; {$I+} exit; end;
with g_AvaHeader
do begin
   viga:=0;
   readln(f,s);  {ID}
   if s<>'WinRaman andmefail' then goto 1;
   inc(viga);
   readln(f,kuup);
   inc(viga);
   readln(f,s);  {algus}
   bl:=StrToLong(s,algus);
   algus:=48*algus;
   if (not bl) then goto 1;
   inc(viga);
   readln(f,s);  {samm}
   bl:=StrToLong(s,samm);
   if ((samm<1) or (samm>9999) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);  {punkte}
   bl:=StrToLong(s,punkte);
   if ((punkte<3) or (punkte>9999) or (not bl)) then goto 1;
   l6pp:=algus+(punkte-1)*samm;
   inc(viga);
   readln(f,s);  {aeg}
   bl:=StrToLong(s,aeg);
   if ((aeg<10) or (aeg>3500) or (not bl)) then goto 1;  {?????}
   aeg:=round(aeg*100/1000);
   inc(viga);
   readln(f,s);  {pinge1}
   bl:=StrToLong(s,pinge1);
   if ((pinge1<0) or (pinge1>2000) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);  {pinge2}
   bl:=StrToLong(s,pinge2);
   if ((pinge2<0) or (pinge2>2000) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);  {myra1}
   bl:=StrToLong(s,myra1);
   if ((myra1<0) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);  {myra2}
   s:=copy(s,18,10);  {string sisaldab numbrit 6  !!!!!}
   bl:=StrToLong(s,myra2);
   if ((myra2<0) or (not bl)) then goto 1;
   inc(viga);
   readln(f,s);  {laser1}
   bl:=StrToLong(s,laser1);
   laser1:=48*laser1;
   if (not bl) then goto 1;
   inc(viga);
   readln(f,s);  {laser2}
   s:=copy(s,20,10);  {string sisaldab numbrit 6  !!!!!}
   bl:=StrToLong(s,laser2);
   laser2:=48*laser2;
   if (not bl) then goto 1;
   inc(viga);
   readln(f,s);  {komm}
   s:=copy(s,13,255);
   StrPCopy(komm,s);
   for i:=1 to 3 do begin inc(viga); readln(f); if IOResult<>0 then goto 1 end;  {!!!!!!}
   max1:=0;
   max2:=0;
   for i:=0 to punkte do begin
                        inc(viga);
                        readln(f,r,g_m1[i], g_m2[i]);
                        if g_m1[i]>g_m1[max1] then max1:=i;
                        if g_m2[i]>g_m2[max2] then max2:=i;
```

```
                        end;
   end;  {with}
Close(f); {$I+}
if IOResult<>0 then begin viga:=-2; goto 1; end;
LoeSisse:=TRUE;
exit;
1: Close(f); {$I+}
end;

procedure M6lise(b: byte);
var i: integer;
    s, s1: string;
begin
case b of 1: begin
             str(g_Reeper1 div 48,s1);
             s:='Spektri algus peab jääma vahemikku '+s1;
             str(g_Reeper2 div 48,s1);
             s:=s+'..'+s1+' cm!'+#0;
             end;
          2: s:='Loendusaeg peab jääma vahemikku 10..9999 ms!'+#0;
          3: begin
             str(g_Reeper1 div 48,s1);
             s:='Spektri l6pp peab jääma vahemikku '+s1;
             str(g_Reeper2 div 48,s1);
             s:=s+'..'+s1+' cm!'+#0;
             end;
          4: s:='Spektri lõpp peab olema algusest suurem vähemalt 1 cm võrra!'+#0;
          5: s:='Skaneerimise samm peab jääma vahemikku 0.01..100 cm!'+#0;
          6: s:='Katsepunkte ei tohi olla üle 5000!'+#0;
          7: s:='Faili nimi on vigane!'+#0;
end;
i:=MessageBox(g_hwMain,@s[1],'Viga sisestamisel:',MB_OK+MB_ICONEXCLAMATION);
end;

procedure Arvuta_Aeg;
var s: string;
    l: longint;
begin
l:=((g_T66Header.l6pp-g_T66Header.algus) div g_T66Header.samm)+1;
str(l,s); s:=s+#0;
SetDlgItemText(g_hwMain,110,@s[1]);
end;

procedure T2idaLahtrid(hw : hWnd; p : pAndmed);
var s1: string;
    r1: real;
begin
with p^
do begin
str(algus div 48,s1); s1:=s1+#0; SetDlgItemText(hw,101,@s1[1]);
r1:=samm/48;
str(r1:2:2,s1); s1:=s1+#0; SetDlgItemText(hw,103,@s1[1]);
l6pp:=algus+samm*(punkte-1);
str(round(l6pp/48),s1); s1:=s1+#0; SetDlgItemText(hw,102,@s1[1]);
str(punkte,s1); s1:=s1+#0; SetDlgItemText(hw,110,@s1[1]);
r1:=1000*aeg/100;
str(round(r1),s1); s1:=s1+#0; SetDlgItemText(hw,104,@s1[1]);
str(pinge1,s1); s1:=s1+#0; SetDlgItemText(hw,108,@s1[1]);
str(pinge2,s1); s1:=s1+#0; SetDlgItemText(hw,109,@s1[1]);
SetDlgItemText(hw,105,@nimi);
SetDlgItemText(hw,106,@laiend);
SetDlgItemText(hw,107,@komm);
end;  {with}
end;

function MainProc(DlgWin: HWND; Msg, WParam, LParam: longint): longint; stdcall;
var i1, i3: integer;
    l1: longint;
    r1: real;
    s1: string;
    p1: pointer;
    TR1: TRect;
    PS1: TPaintStruct;
    buf1, buf2: array[0..255] of char;
label 1;
begin
MainProc:=1;
```

```
case Msg
of WM_PAINT: begin
             MainProc:=0;
             if GetUpdateRect(DlgWin,TR1,FALSE)
             then begin
                  if (g_Sodi.yks and IntersectRect(TR1,g_TRGraafik,TR1))
                  then g_Sodi.yks:=FALSE
                  end
             else if g_Sodi.yks
                  then InvalidateRect(DlgWin,@g_TRGraafik,FALSE)
                  else exit;
             BeginPaint(DlgWin,PS1);
             {}
             SetBkMode(PS1.HDC,TRANSPARENT);
             SelectObject(PS1.HDC,g_Font1);
             if IntersectRect(TR1,g_TRxtelg,PS1.rcPaint)
             then with g_T66Header
                  do begin  {x-telg}
                     SelectObject(PS1.HDC,g_MustSulg);
                     MoveToEx(PS1.HDC,20,301,nil);
                     LineTo(PS1.HDC,550,301);
                     SetTextAlign(PS1.HDC,TA_CENTER+TA_TOP);
                     SelectObject(PS1.HDC,g_HallSulg);
                     for i3:=0 to xnr
                     do begin
                        MoveToEx(PS1.HDC,20+round(dx5*i3),301,nil);
                        LineTo(PS1.HDC,20+round(dx5*i3),304);
                        if g_M66tmiseTyyp=1
                        then begin
                             r1:=x0+i3*dx2;
                             s1:=L6iguReal(r1,1);
                             TextOut(PS1.HDC,20+round(i3*dx5),306,@s1[1],length(s1)-1);
                             end;
                        end
                     end;  {with}
             {}
             if IntersectRect(TR1,g_TRGraafik,PS1.rcPaint)
             then begin
                  SelectClipRgn(PS1.HDC,g_HRGraafik);
                  if g_Sodi.yks
                  then with g_T66Header
                       do begin  {yks punkt}
                          SelectObject(PS1.HDC,g_MustSulg);
                          {PS1.rcPaint:=g_TRGraafik;}  {!!!!!}
                          r1:=k1*g_m1[g_Punkt-1];
                          if r1>30000 then r1:=30000;
                          MoveToEx(PS1.HDC,20+round(dx1*(g_Punkt-1)),200-round(r1),nil);
                          r1:=k1*g_m1[g_Punkt];
                          if r1>30000 then r1:=30000;
                          LineTo(PS1.HDC,20+round(dx1*g_Punkt),200-round(r1));
                          g_Sodi.yks:=FALSE;
                          end
                  else with g_T66Header
                       do begin  {kogu graafik ymber joonistada}
                          FillRect(PS1.HDC,g_TRGraafik,g_HallPintsel);  {!!!!!}
                          SelectObject(PS1.HDC,g_HallSulg);
                          for i3:=1 to 10
                          do begin  {horisontaalsed kriipsud}
                             MoveToEx(PS1.HDC,20,300-16*i3,nil);
                             LineTo(PS1.HDC,550,300-16*i3);
                             end;
                          for i3:=0 to xnr
                          do begin  {vertikaalsed kriipsud}
                             MoveToEx(PS1.HDC,20+round(dx5*i3),140,nil);
                             LineTo(PS1.HDC,20+round(dx5*i3),300);
                             end;
                          {}
                          SelectObject(PS1.HDC,g_MustSulg);
                          r1:=k1*g_m1[0];
                          if r1>30000 then r1:=30000;
                          MoveToEx(PS1.HDC,20,300-round(r1),nil);
                          for i3:=1 to g_Sodi.nr
                          do begin
                             r1:=k1*g_m1[i3];
                             if r1>30000 then r1:=30000;
                             LineTo(PS1.HDC,20+round(dx1*i3),300-round(r1));
                             end;
```

```
                    end;  {else with}
                SelectClipRgn(PS1.HDC,0);
                end;
            {}
            EndPaint(DlgWin,PS1);
            exit;
            end;  {WM_PAINT}
WM_COMMAND: begin
              case wParam
              of 101: begin  {Algus}
                      if SendDlgItemMessage(DlgWin,101,EM_GETMODIFY,0,0)<>0
                      then begin
                           SendDlgItemMessage(DlgWin,101,EM_SETMODIFY,0,0);
                           l1:=48*GetDlgItemInt(DlgWin,101,g_bl,FALSE);
                           if ((g_T66Header.algus=l1) or (not g_bl) or (l1>g_T66Header.l6pp)
                               or (l1<g_Reeper1) or (l1>g_Reeper2)) then exit;
                           g_T66Header.algus:=l1;
                           Arvuta_Aeg;
                           end;
                      exit;
                      end;
                 102: begin  {L6pp}
                      if SendDlgItemMessage(DlgWin,102,EM_GETMODIFY,0,0)<>0
                      then begin
                           SendDlgItemMessage(DlgWin,102,EM_SETMODIFY,0,0);
                           l1:=48*GetDlgItemInt(DlgWin,102,g_bl,FALSE);
                           if ((g_T66Header.l6pp=l1) or (not g_bl)
                               or (l1<(g_T66header.algus+48)) or (l1>g_Reeper2)) then exit;
                           g_T66Header.l6pp:=l1;
                           Arvuta_Aeg;
                           end;
                      exit;
                      end;
                 103: begin  {Samm}
                      if SendDlgItemMessage(DlgWin,103,EM_GETMODIFY,0,0)<>0
                      then begin
                           SendDlgItemMessage(DlgWin,103,EM_SETMODIFY,0,0);
                           l1:=GetDlgItemText(DlgWin,103,@buf1,255);
                           g_bl:=TextToFloat(@buf1,255,r1);
                           l1:=round(48*r1);
                           if ((not g_bl) or (l1<1) or (l1>9999) or (l1=g_T66Header.samm))
                           then exit;
                           g_T66Header.samm:=l1;
                           Arvuta_Aeg;
                           end;
                      exit;
                      end;
                 111: begin  {yhel koordinaadil}
                      if SendDlgItemMessage(DlgWin,111,BM_GETCHECK,0,0)=BST_CHECKED
                      then begin
                           EnableWindow(GetDlgItem(DlgWin,102),FALSE);
                           EnableWindow(GetDlgItem(DlgWin,103),FALSE);
                           EnableWindow(GetDlgItem(DlgWin,105),FALSE);
                           EnableWindow(GetDlgItem(DlgWin,106),FALSE);
                           EnableWindow(GetDlgItem(DlgWin,107),FALSE);
                           EnableWindow(GetDlgItem(DlgWin,110),FALSE);
                           EnableWindow(GetDlgItem(DlgWin,112),FALSE);
                           end
                      else begin
                           EnableWindow(GetDlgItem(DlgWin,102),TRUE);
                           EnableWindow(GetDlgItem(DlgWin,103),TRUE);
                           EnableWindow(GetDlgItem(DlgWin,105),TRUE);
                           EnableWindow(GetDlgItem(DlgWin,106),TRUE);
                           EnableWindow(GetDlgItem(DlgWin,107),TRUE);
                           EnableWindow(GetDlgItem(DlgWin,110),TRUE);
                           EnableWindow(GetDlgItem(DlgWin,112),TRUE);
                           end;
                      exit;
                      end;
                 112: begin  {Failist...}
                      if OpenFileName(1,@buf1)
                      then begin
                           if LoeSisse(@buf1,g_i1)
                           then begin
                                Move(g_AvaHeader,g_T66Header,SizeOf(andmed));
                                g_T66Header.nimi:='';
                                Move(g_DefaultHeader.laiend,g_T66Header.laiend,5);
```

```
                T2idaLahtrid(DlgWin,@g_T66Header);
                SendDlgItemMessage(DlgWin,101,EM_SETSEL,0,-1);
                SetFocus(getdlgitem(DlgWin,101));
                exit;
                end
           else begin
                str(g_i1,s1);
                s1:='Fail on vigane, kood '+s1+#0;
                i1:=MessageBox(DlgWin,@s1[1],'Viga:',MB_OK+MB_ICONEXCLAMATION);
                exit;
                end;
           end
      else exit;
      end;
  113: begin  {Sulge/Katkesta}
      GetDlgItemText(DlgWin,113,@buf1,10);
      i1:=StrIComp(@buf1,'Katkesta');
      if i1=0
      then begin  {hakkab katkestama}
           i1:=MessageBox(DlgWin,'Kas t6esti katkestan?','',
                          MB_YESNO+MB_ICONEXCLAMATION);
           if i1<>IDYES then exit;
           if g_Init
           then begin  {toimub initsialiseerimine}
                exit;
                end
           else begin  {toimub m66tmine}
                salvesta('raman.tmp');
                Status('Katkestan m66tmise. Sellega võib minna aega...');
                g_Katkestan:=TRUE;
                SetDlgitemText(DlgWin,113,'Sulge');
                exit;
                end;
           end
      else begin  {nupp oli Sulge}
           PostMessage(DlgWin,WM_CLOSE,0,0);
           exit;
           end;
      end;
  114: begin  {Abi}
      i1:=MessageBox(DlgWin,'Programm DFS-52 juhtimiseks','About:',MB_OK);
      exit;
      end;
  115: begin  {M66da}
      with g_T66Header
      do begin
         algus:=GetDlgItemInt(DlgWin,101,g_bl,FALSE);
         algus:=48*algus;
         if ((not g_bl) or (algus<g_Reeper1) or (algus>g_Reeper2))
         then begin
              M6lise(1);
              SendDlgItemMessage(DlgWin,101,EM_SETSEL,0,-1);
              SetFocus(GetDlgItem(DlgWin,101));
              exit;
              end;
         aeg:=GetDlgItemInt(DlgWin,104,g_bl,FALSE);
         if ((not g_bl) or (aeg<10) or (aeg>9999))
         then begin
              M6lise(2);
              SendDlgItemMessage(DlgWin,104,EM_SETSEL,0,-1);
              SetFocus(GetDlgItem(DlgWin,104));
              exit;
              end;
         aeg:=round(aeg*100/1000);
         if SendDlgItemMessage(DlgWin,111,BM_GETCHECK,0,0)<>BST_CHECKED
         then begin  {tavaline m66tmine}
              g_M66tmiseTyyp:=1;
              l6pp:=GetDlgItemInt(DlgWin,102,g_bl,FALSE);
              if ((not g_bl) or (l6pp<(g_Reeper1 div 48))
                  or (l6pp>(g_Reeper2 div 48)))
              then begin
                   M6lise(3);
                   SendDlgItemMessage(DlgWin,102,EM_SETSEL,0,-1);
                   SetFocus(GetDlgItem(DlgWin,102));
                   exit;
                   end;
              l6pp:=48*l6pp;
```

```
                if (l6pp<algus+48)
                then begin
                     M6lise(4);
                     SendDlgItemMessage(DlgWin,102,EM_SETSEL,0,-1);
                     SetFocus(GetDlgItem(DlgWin,102));
                     exit;
                     end;
                i1:=GetDlgItemText(DlgWin,103,@buf1,255);
                g_bl:=TextToFloat(@buf1,255,r1);
                r1:=48*r1;
                samm:=trunc(r1);
                if ((not g_bl) or (samm<1) or (samm>9999))
                then begin
                     M6lise(5);
                     SendDlgItemMessage(DlgWin,103,EM_SETSEL,0,-1);
                     SetFocus(GetDlgItem(DlgWin,103));
                     exit;
                     end;
                punkte:=((l6pp-algus) div samm)+1;
                if punkte>5000
                then begin
                     M6lise(6);
                     SendDlgItemMessage(DlgWin,102,EM_SETSEL,0,-1);
                     SetFocus(GetDlgItem(DlgWin,102));
                     exit;
                     end;
                i1:=GetDlgItemText(DlgWin,105,@nimi,50);  {nimi}
                if (not OnNimi(nimi))
                then begin
                     M6lise(7);
                     SendDlgItemMessage(DlgWin,105,EM_SETSEL,0,-1);
                     SetFocus(GetDlgItem(DlgWin,105));
                     exit;
                     end;
                Move(g_DefaultHeader.laiend,g_T66Header.laiend,5);  {?????}
                i1:=GetDlgItemText(DlgWin,107,@komm,255);  {komm}
                end
           else g_M66tmiseTyyp:=2;  {yhel koordinaadil}
           end;  {with}
        EnableWindow(GetDlgItem(DlgWin,112),FALSE);
        EnableWindow(GetDlgItem(DlgWin,115),FALSE);
        PostMessage(DlgWin,WM_COMMAND,602,0);  {hakkab m66tma}
        exit;
        end;
    120: begin  {-}
        g_T66Header.k1:=g_T66Header.k1/1.2;
        if g_T66Header.k1<g_T66Header.k1min
        then g_T66Header.k1:=g_T66Header.k1min;
        g_T66Header.k1max:=g_T66Header.k1*1000;  {?????}
        g_T66Header.k1min:=g_T66Header.k1/5;  {?????}
        g_Sodi.yks:=FALSE;
        InvalidateRect(DlgWin,@g_TRGraafik,TRUE);
        exit;
        end;
    121: begin  {+}
        g_T66Header.k1:=g_T66Header.k1*1.2;
        if g_T66Header.k1>g_T66Header.k1max
        then g_T66Header.k1:=g_T66Header.k1max;
        g_T66Header.k1max:=g_T66Header.k1*1000;  {?????}
        g_T66Header.k1min:=g_T66Header.k1/5;  {?????}
        g_Sodi.yks:=FALSE;
        InvalidateRect(DlgWin,@g_TRGraafik,TRUE);
        exit;
        end;
        {*****  Siit algavad threadide teated  *****}
    501: begin {T66Proc0 saadab}
        case lParam
        of 0: begin  {k6ik korras}
              g_Init:=FALSE;
              SetDlgItemText(DlgWin,113,'Sulge');
              EnableWindow(GetDlgItem(DlgWin,115),TRUE);
              Status('Sisesta mõõtmise parameetrid ja pressi "Mõõda"!');
              exit;
              end;
           1: begin  {PIC ei vasta}
              Status('Aparatuur ei vasta... Kas plokid on sisse lülitatud?');
              ResetPort;
```

```
          piip;
          i1:=MessageBox(DlgWin,'Aparatuur ei vasta!',
                         'Viga:',MB_RETRYCANCEL+MB_ICONEXCLAMATION);
          if i1=IDRETRY
          then begin
               g_l1:=DlgWin;  {l1 peab olema globaalne}
               g_htr1:=CreateThread(nil,0,@T66Proc0,@g_l1,0,dw1);  {Init}
               Status('Initsialiseerimine...');
               exit;
               end
          else begin
               ResetPort;
               SulePort;
               SendMessage(DlgWin,WM_CLOSE,0,0);
               exit;
               end;
          end;
       3: begin  {kunstlik l6petamine valmis}
          Status('Katkestatud');
          SulePort;
          exit;
          end
     end;  {case}
     end;
502: begin  {T66Proc1 saadab: yks punkt j@lle m66detud}
     inc(g_Punkt);
     if ((g_Punkt>560) and (g_M66tmiseTyyp=2)) then g_Punkt:=0;
     str(g_Punkt,s1);
     s1:=s1+#0;
     SetDlgItemText(DlgWin,123,@s1[1]);
     with g_T66Header
     do begin
        if g_m1[g_Punkt]>g_m1[max1] then max1:=g_Punkt;
        if (k1*g_m1[max1]<=160)
        then begin  {yks punkt}
             g_Sodi.nr:=g_Punkt;
             g_Sodi.yks:=TRUE;
             PostMessage(DlgWin,WM_PAINT,0,0);
             end
        else begin  {vaja mastaapi muuta}
             k1:=160/g_m1[g_Punkt];
             k1max:=k1*1000;  {?????}
             k1min:=k1/5;  {?????}
             g_Sodi.nr:=g_Punkt;
             g_Sodi.yks:=FALSE;
             InvalidateRect(DlgWin,@g_TRGraafik,FALSE);
             exit;
             end;
        end;  {with}
     exit;
     end;
503: begin  {T66Proc1 l6ppenud, m66tmine valmis}
     case g_M66tmiseTyyp
     of 1: begin
           Status('Mõõdan pimevoolu...');
           g_l1:=DlgWin;  {l1 peab olema globaalne}
           g_htr1:=CreateThread(nil,0,@T66Proc2,@g_l1,0,dw1);  {myra m66tmine}
           end;
        2: begin
           PostMessage(DlgWin,WM_COMMAND,507,2);
           end;
     end;  {case}
     exit;
     end;
504: begin  {T66Proc3 l6pp, k6ik valmis}
     Status('Mõõtmine on lõppenud');
     SetDlgItemText(DlgWin,105,'');  {Nimi}
     for i1:=101 to 105 do EnableWindow(GetDlgItem(DlgWin,i1),TRUE);
     EnableWindow(GetDlgItem(DlgWin,107),TRUE);  {Kommentaar}
     EnableWindow(GetDlgItem(DlgWin,111),TRUE);  {Yhel koordinaadil}
     EnableWindow(GetDlgItem(DlgWin,112),TRUE);  {Failist...}
     EnableWindow(GetDlgItem(DlgWin,115),TRUE);  {M66da}
     exit;
     end;
506: begin  {T66Proc4 l6pp, on koordinaadil}
     case g_M66tmiseTyyp
     of 1: begin
```

```
            Status('Mõõdan pimevoolu...');
            g_M66danMyra1:=TRUE;
            g_l1:=DlgWin;  {l1 peab olema globaalne}
            g_htr1:=CreateThread(nil,0,@T66Proc2,@g_l1,0,dw1);  {myra m66tmine}
            end;
         2: begin
            PostMessage(DlgWin,WM_COMMAND,507,1);
            end;
      end;  {case}
      exit;
      end;
  507: begin  {myra1 m66tmine}
      case loword(lParam)
      of 0: begin  {yks punkt m66detud}
            str(hiword(lParam),s1);
            Status2s(s1+'/10');
            exit;
            end;
         1: begin  {myra1 valmis}
            g_M66danMyra1:=FALSE;
            Status('Mõõdan... Katkestamiseks pressi "Katkesta"');
            Status2s('');
            g_l1:=DlgWin;  {l1 peab olema globaalne}
            g_htr1:=CreateThread(nil,0,@T66Proc1,@g_l1,0,dw1);  {M66tmine}
            exit;
            end;
         2: begin  {myra2 valmis}
            Status('Salvestan...');
            Status2s('');
            StrLCopy(@buf1,g_DataPath,255);
            LStrCat(@buf1,g_T66Header.nimi);
            LStrCat(@buf1,g_T66Header.laiend);
            salvesta(@buf1);
            salvesta('raman.tmp');
            SetDlgitemText(DlgWin,113,'Sulge');
            buf1:='Mõõtmine on lõppenud ja tulemus salvestatud faili ';
            LStrCat(@buf1,g_T66Header.nimi);
            LStrCat(@buf1,g_T66Header.laiend);
            {Status(buf1);}
            SetDlgItemText(DlgWin,105,'');
            for i1:=101 to 105 do EnableWindow(GetDlgItem(DlgWin,i1),TRUE);
            EnableWindow(GetDlgItem(DlgWin,107),TRUE);
            EnableWindow(GetDlgItem(DlgWin,111),TRUE);
            EnableWindow(GetDlgItem(DlgWin,115),TRUE);  {M66da}
            exit;
            end
         else exit
      end;  {case}
      end;
  509: begin  {jama}
      case lParam
      of 0: begin  {pordi jama, fataalne}
            i1:=MessageBox(DlgWin,'Port ei tööta!','Fataalne viga:',
                           MB_OK+MB_ICONEXCLAMATION);
            PostMessage(DlgWin,WM_COMMAND,509,3);
            exit;
            end;
         1: begin  {PIC ei vasta}
            i1:=MessageBox(DlgWin,'Aparatuur ei vasta!'+#13+
                           'Vaata, et plokid oleksid sisse lülitatud!'
                           +#13+'Kas proovin uuesti?',
                           'Viga:',MB_RETRYCANCEL+MB_ICONEXCLAMATION);
            if ((i1=IDRETRY){ and ResetPort})
            then begin  {alustab uuesti}
                 PostMessage(DlgWin,WM_COMMAND,505,0);
                 exit;
                 end;
            PostMessage(DlgWin,WM_COMMAND,509,3);  {l6pp}
            exit;
            end;
         3: begin  {kunstlik l6petamine}
            g_Katkestan:=FALSE;
            g_l1:=DlgWin;  {l1 peab olema globaalne}
            g_htr1:=CreateThread(nil,0,@T66Proc3,@g_l1,0,dw1);  {l6petamine}
            exit;
            end;
      end;  {case}
```

```
                    end;
                601: begin  {initsialiseerimine}
                    SendDlgItemMessage(DlgWin,101,EM_SETSEL,0,-1);
                    SetFocus(getdlgitem(DlgWin,101));
                    if (AvaPort and ResetPort)
                    then begin
                        g_Init:=TRUE;
                        EnableWindow(GetDlgItem(DlgWin,113),TRUE);  {Katkesta}
                        PostMessage(DlgWin,WM_COMMAND,505,0);  {initsialiseerimine}
                        g_l1:=DlgWin;  {l1 peab olema globaalne}
                        g_htr1:=CreateThread(nil,0,@T66Proc0,@g_l1,0,dw1);  {Init}
                        Status('Initsialiseerimine...');
                        exit;
                        end
                    else begin  {port ei avane}
                        i1:=MessageBox(DlgWin,'COM-port ei avane!',
                                       'Viga:',MB_OK+MB_ICONEXCLAMATION);
                        PostMessage(DlgWin,WM_CLOSE,0,0);
                        exit;
                        end;
                    end;
                602: begin  {M66tmise alustamine}
                    for i1:=101 to 105 do EnableWindow(GetDlgItem(DlgWin,i1),FALSE);
                    EnableWindow(GetDlgItem(DlgWin,107),FALSE);
                    EnableWindow(GetDlgItem(DlgWin,111),FALSE);
                    CheckDlgButton(DlgWin,111,0);  {Yhel koordinaadil}
                    with g_T66Header  {graafiku parameetrite arvutamine}
                    do begin
                       case g_M66tmiseTyyp
                       of 1: begin  {p6hiline m66tmine}
                             dx1:=560/(punkte-1);  {x-nihe pikslites}
                             r1:=(samm*(punkte-1))/48;
                             dx5:=560/r1;  {pikslit nanomeetrile}
                             r1:=ln(r1)/ln(10);  {log(l6pp-algus)}
                             if r1>=0 then r1:=trunc(r1) else r1:=trunc(r1)-1;
                             {ceil ei t66ta}
                             dx2:=exp(ln(10)*r1);  {jaotise suurus}
                             r1:=(samm*(punkte-1))/(48*dx2);
                             xnr:=trunc(r1);  {jaotiste arv}
                             if xnr<2 then dx2:=dx2/5 else if xnr<5 then dx2:=dx2/2;
                             r1:=(samm*(punkte-1))/(48*dx2);
                             xnr:=round(r1);  {jaotiste arv}
                             dx5:=dx5*dx2;  {jaotise suurus}
                             x0:=algus div 48;
                             str(punkte,s1); s1:=s1+#0; SetDlgItemText(DlgWin,110,@s1[1]);
                             end;  {p6hiline m66tmine}
                          2: begin  {m66tmine pysival koordinaadil}
                             Status('Mõõtmine püsival koordinaadil');
                             samm:=0;
                             punkte:=600;  {?????}
                             dx1:=1;
                             end;
                       end;  {case}
                       Status('Liigun koordinaadile...');
                       g_l1:=DlgWin;  {l1 peab olema globaalne}
                       g_htr1:=CreateThread(nil,0,@T66Proc4,@g_l1,0,dw1);
                       {koordinaadile minek}
                       SetDlgItemText(DlgWin,113,'Katkesta');
                       g_Punkt:=-1;
                       InvalidateRect(DlgWin,nil,TRUE);
                       PostMessage(DlgWin,WM_PAINT,0,0);
                       exit;
                       end;  {with}
                    end  {602}
                else begin MainProc:=0; exit; end;
                end; {case wParam}
                end;  {WM_COMMAND}
WM_INITDIALOG: begin
                g_hwMain:=DlgWin;
                GetCurrentDirectory(255,@buf1);
                p1:=lstrcat(@buf1,'\raman_95.ini');
                i1:=GetPrivateProfileString('GENERAL','DATAPATH','',@g_DataPath,255,p1);
                g_DataPath[i1]:='\';
                g_DataPath[i1+1]:=#0;
                with g_DefaultHeader
                do begin
                   g_Reeper1:=48*GetPrivateProfileInt('GENERAL','REEPER1',1000,p1);
```

```
                    g_Reeper2:=48*GetPrivateProfileInt('GENERAL','REEPER2',1000,p1);
                    i1:=GetPrivateProfileString('GENERAL','FAILINIMI','mdr_95',@nimi,10,p1);
                    i1:=GetPrivateProfileString('GENERAL','LAIEND','.dat',@laiend[1],10,p1);
                    laiend[0]:='.';
                    algus:=48*GetPrivateProfileInt('DEFAULT','ALGUS',401,p1);
                    punkte:=GetPrivateProfileInt('DEFAULT','PUNKTE',701,p1);
                    i1:=GetPrivateProfileString('DEFAULT','SAMM',nil,@buf2,255,p1);
                    TextToFloat(@buf2,255,r1);
                    samm:=round(40*r1);
                    if ((samm<3) or (samm>9999)) then samm:=48;
                    l6pp:=algus+samm*(punkte-1);
                    aeg:=GetPrivateProfileInt('DEFAULT','LOENDUSAEG',101,p1);
                    pinge1:=GetPrivateProfileInt('DEFAULT','PINGE1',1000,p1);
                    pinge2:=GetPrivateProfileInt('DEFAULT','PINGE2',1000,p1);
                    laser1:=48*GetPrivateProfileInt('DEFAULT','RAYLEIGH1',1000,p1);
                    laser2:=48*GetPrivateProfileInt('DEFAULT','RAYLEIGH2',1000,p1);
                    i1:=GetPrivateProfileString('DEFAULT','KOMMENTAAR','ff',@komm,255,p1);
                    end;
                  SetWindowText(DlgWin,'Spektrograafi DFS-52 juhtprogramm');
                  Status('Initsialiseerimine...');
                  g_Sodi.nr:=0;
                  g_Sodi.yks:=FALSE;  {?????}
                  g_Katkestan:=FALSE;
                  g_x:=0;
                  T2idaLahtrid(DlgWin,@g_DefaultHeader);
                  {}
                  g_Font1:=GetStockObject(ANSI_VAR_FONT);
                  g_MustSulg:=GetStockObject(BLACK_PEN);
                  g_HallSulg:=CreatePen(PS_SOLID,0,$009F9F9F);
                  g_PunaneSulg:=CreatePen(PS_SOLID,0,$000000FF);
                  g_HallPintsel:=GetStockObject(LTGRAY_BRUSH);
                  with g_TRGraafik do begin left:=20; top:=140; right:=551; bottom:=301; end;
                  g_HRGraafik:=CreateRectRgnIndirect(g_TRGraafik);
                  with g_TRxtelg do begin left:=5; top:=301; right:=560; bottom:=320; end;
                  g_Filter:=FALSE;
                  CheckDlgButton(DlgWin,111,0);  {yhel koordinaadil}
                  EnableWindow(GetDlgItem(DlgWin,115),FALSE);  {M66da}
                  SetDlgItemText(DlgWin,113,'Katkesta');
                  EnableWindow(GetDlgItem(DlgWin,113),FALSE);  {Sulge/Katkesta}
                  Move(g_DefaultHeader,g_T66Header,SizeOf(andmed));
                  SetWindowPos(DlgWin,0,(GetSystemMetrics(SM_CXSCREEN)-620) shr 1,
                               (GetSystemMetrics(SM_CYSCREEN)-440)shr 1,0,0,
                               SWP_NOZORDER+SWP_NOSIZE);
                  PostMessage(DlgWin,WM_COMMAND,601,0);  {initsialiseerimine}
                  exit;
                1: i1:=MessageBox(DlgWin,'ini-fail on vigane!','Viga:',MB_OK+MB_ICONEXCLAMATION);
                  EndDialog(DlgWin,0);
                  end;  {WM_INITDIALOG}
   WM_CLOSE: begin
             MainProc:=0;
             SulePort;
             DeleteObject(g_Font1);
             DeleteObject(g_HallPintsel);
             DeleteObject(g_HallSulg);
             DeleteObject(g_PunaneSulg);
             DeleteObject(g_MustSulg);
             DeleteObject(g_HRGraafik);
             EndDialog(DlgWin,0);
             exit;
             end
   else begin MainProc:=0; exit; end;
end; {case Message}
end;  {MainProc}

begin
g_i1:=DialogBox(hInstance,'M66TMINE',0,@MainProc);
end.
```

Windows-programmi ressursiskript

```
/************************************************************************

                              R95RUN.RC

************************************************************************/

M66TMINE DIALOG 50, 25, 380, 230
STYLE WS_OVERLAPPED | WS_VISIBLE | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX
CAPTION "M66dan..."
FONT 8, "MS Sans Serif"
LANGUAGE LANG_NEUTRAL, SUBLANG_NEUTRAL
{
 CONTROL "Sulge", 113, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         255, 17, 37, 12
 CONTROL "-", 120, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 309, 200, 8, 8
 CONTROL "+", 121, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 317, 200, 8, 8
 CONTROL "Mxxda", 115, "BUTTON", BS_DEFPUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         255, 43, 37, 12
 CONTROL "Abi", 114, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         255, 30, 37, 12
 CONTROL "22222", 101, "EDIT", ES_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_TABSTOP,
         73, 5, 24, 10
 CONTROL "22222", 103, "EDIT", ES_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_TABSTOP,
         73, 27, 24, 10
 CONTROL "22222", 104, "EDIT", ES_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_TABSTOP,
         73, 38, 24, 10
 CONTROL "ABBBBBBA", 105, "EDIT", ES_RIGHT | ES_AUTOHSCROLL | WS_CHILD | WS_VISIBLE
         | WS_BORDER | WS_TABSTOP, 55, 49, 42, 10
 CONTROL "blaa-blaa", 107, "EDIT", ES_LEFT | ES_AUTOHSCROLL | WS_CHILD | WS_VISIBLE
         | WS_BORDER | WS_TABSTOP, 55, 60, 249, 10
 CONTROL "22222", 102, "EDIT", ES_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_TABSTOP,
         73, 16, 24, 10
 CONTROL "Mxxda ьhel koordinaadil", 111, "BUTTON", BS_AUTOCHECKBOX | WS_CHILD | WS_VISIBLE
         | WS_TABSTOP, 140, 42, 90, 8
 CONTROL "Failist...", 112, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP,
         255, 4, 37, 12
 LTEXT "Status", 122, 5, 217, 370, 12, WS_BORDER
/* RTEXT "00:00:00", 123, 274, 205, 29, 8 */
 RTEXT "Spektri lxpp:", -1, 10, 17, 45, 8
 RTEXT "Samm:", -1, 10, 28, 45, 8
 RTEXT "9999", 110, 206, 28, 16, 8
 RTEXT "2000", 108, 204, 7, 16, 8
 RTEXT "Spektri algus:", -1, 10, 6, 45, 8
 RTEXT "2000", 109, 204, 17, 16, 8
 RTEXT "I FEK anoodpinge:", -1, 140, 7, 62, 8
 RTEXT "Loendusaeg:", -1, 10, 39, 45, 8
 RTEXT "II FEK anoodpinge:", -1, 140, 17, 62, 8
 RTEXT "Katsepunkte:", -1, 140, 27, 50, 8
 RTEXT "Faili nimi:", -1, 10, 50, 45, 8
 RTEXT "Kommentaar:", -1, 10, 61, 45, 8
 RTEXT ".dat", 106, 99, 50, 14, 8
 RTEXT "cm", -1, 99, 6, 10, 8
 RTEXT "V", -1, 222, 7, 5, 8
 RTEXT "cm", -1, 99, 17, 10, 8
 RTEXT "cm", -1, 99, 28, 10, 8
 RTEXT "ms", -1, 99, 39, 10, 8
 RTEXT "V", -1, 222, 17, 5, 8
 RTEXT "-1", -1, 109, 3, 5, 8
 RTEXT "-1", -1, 109, 14, 5, 8
 RTEXT "-1", -1, 109, 25, 5, 8
}
```

Windows-programmi ini-fail

```
[GENERAL]
DATAPATH=C:\PROGRA~1\WINRAMAN\DATA
FAILINIMI=raman_95
LAIEND=dat
PORT=com1
BAUD=9600
REEPER1=16690
REEPER2=25000

[DEFAULT]
ALGUS=16700
PUNKTE=11
SAMM=48
LOENDUSAEG=50
PINGE1=1600
PINGE2=1300
RAYLEIGH1=18790
RAYLEIGH2=18820
KOMMENTAAR=blaa-blaa
```

Andmefaili struktuur

```
WinRaman andmefail
02.02.2002 16:18
Algus: 17900 cm-1
Samm: 48 /48 cm-1
Katsepunkte: 1801
Loendusaeg: 500 ms
I FEK anoodpinge: 1600V
II FEK anoodpinge: 1300V
Pimeimpulsse alguses: 18
Pimeimpulsse l6pus: 17
Rayleigh' joone algus: 18790 cm-1
Rayleigh' joone l6pp: 18820 cm-1
Kommentaar: Mitt ja K2pp



17900.00 22 0
17901.00 52 0
17902.00 28 0
17903.00 34 0
17904.00 28 0
17905.00 34 0
17906.00 20 0
17907.00 40 0
17908.00 34 0
17909.00 34 0
17910.00 24 0
17911.00 28 0
17912.00 34 0
17913.00 28 0
17914.00 32 0
17915.00 28 0
   ...   ...
```

Lisa III
Mõõtekompleks spektrijoone ülipeenstruktuuri uurimiseks

Windows-keskkonna juhtprogramm

```
program fabry95;

{fp:}
(*{$APPTYPE GUI}
uses Windows, SysUtils;*)
{Delphi:}
uses Windows, Messages, SysUtils, CommDlg, CommCtrl;

type andmed = record
              pilte, ridu, punkte: longint;
              l1, l2, l3: longint;
              dx11, dx21, dx31, dx51: real;  {graafikute koordinaadid}
              dx12, dx22, dx32: real;  {graafikute koordinaadid}
              ky1, ky2, ky3, ky4: real;  {mastaabikordajad}
              nih1: byte;  {mastaabikordaja}
              k0, k1, k2, k3, k4, k5, k6, k7, k8: real;  {polynoomi kordajad}
              max1, max2, max3: integer;
              r1, r2, r3, r4, r5: real;
              aeg: array[0..50] of char;  {!!!!!}
              komm: array[0..255] of char;
              nimi, laiend: array[0..50] of char
              end;
     pAndmed = ^andmed;
     TCAPTUREPARMS = record
                     dwRequestMicroSecPerFrame   : DWORD;  {Requested capture rate}
                     fMakeUserHitOKToCapture     : BOOL;   {Show "Hit OK to cap" dlg?}
                     wPercentDropForError        : UINT;   {Give error msg if > (10%)}
                     fYield                      : BOOL;   {Capture via background task?}
                     dwIndexSize                 : DWORD;  {Max index size in frames (32K)}
                     wChunkGranularity           : UINT;   {Junk chunk granularity (2K)}
                     fUsingDOSMemory             : BOOL;   {Use DOS buffers?}
                     wNumVideoRequested          : UINT;   {# video buffers, If 0, autocalc}
                     fCaptureAudio               : BOOL;   {Capture audio?}
                     wNumAudioRequested          : UINT;   {# audio buffers, If 0, autocalc}
                     vKeyAbort                   : UINT;   {Virtual key causing abort}
                     fAbortLeftMouse             : BOOL;   {Abort on left mouse?}
                     fAbortRightMouse            : BOOL;   {Abort on right mouse?}
                     fLimitEnabled               : BOOL;   {Use wTimeLimit?}
                     wTimeLimit                  : UINT;   {Seconds to capture}
                     fMCIControl                 : BOOL;   {Use MCI video source?}
                     fStepMCIDevice              : BOOL;   {Step MCI device?}
                     dwMCIStartTime              : DWORD;  {Time to start in MS}
                     dwMCIStopTime               : DWORD;  {Time to stop in MS}
                     fStepCaptureAt2x            : BOOL;   {Perform spatial averaging 2x}
                     wStepCaptureAverageFrames   : UINT;   {Temporal average n Frames}
                     dwAudioBufferSize           : DWORD;  {Size of audio bufs (0 = default)}
                     fDisableWriteCache          : BOOL;   {Attempt to disable write cache}
                     AVStreamMaster              : UINT;   {Which stream controls length?}
                     end;
     VIDEOHDR = record
                lpData               : PBYTE;                {pointer to locked data buffer}
                dwBufferLength       : DWORD;                {Length of data buffer}
                dwBytesUsed          : DWORD;                {Bytes actually used}
                dwTimeCaptured       : DWORD;                {Milliseconds from start of stream}
                dwUser               : DWORD;                {for client's use}
                dwFlags              : DWORD;                {assorted flags (see defines)}
                dwReserved           : array[0..3] of DWORD; {reserved for driver}
                end;
     PVIDEOHDR = ^VIDEOHDR;
var g_hwMain, g_hwCap, g_hwClip, g_hwStatus, g_hwKohe: HWND;
    g_hwm: array[1..13] of HWND;
    g_DataPath, g_MyraFail: array[0..255] of char;
    g_T66Header, g_DefaultHeader, g_AvaHeader: andmed;
    g_i1, g_Punkt, g_Rida1, g_Rida2: integer;
    g_x1, g_x2, g_x3, g_x4, g_x8: integer;
    g_y1, g_y2, g_y3, g_y4, g_y5, g_y6, g_y8: integer;
```

```
    g_dx1, g_dx2, g_dx3, g_dx4, g_dx8: integer;
    g_dy1, g_dy2, g_dy3, g_dy4, g_dy5, g_dy6, g_dy7, g_dy8: integer;
    g_dy11, g_dy21: real;  {jaotise k6rgus pikslites}
    g_Olukord, g_Heledus: byte;
    g_s1: string;
    g_MaxX, g_MaxY: integer;
    g_m1, g_m2, g_m3: array[1..999] of longint;
    g_Buf1: array[1..999] of word;  {1 rida}
    g_hm1: hmenu;
    g_hf0, g_hf1: hFont;
    g_hp0, g_hp1, g_hp2, g_hp3, g_hp4: hPen;
    g_hb0, g_hb1, g_hb2, g_hb3: hBrush;
    g_DCMain: hDC;
    g_hBuf1: tHandle;
    g_pBuf1: pointer;
    g_Punkt1: tPoint;
    bmi1: TBitmapInfo;
    WindowClass, wclass2: TWndClass;
    g_TR1: TRect;
    g_Sehkendan: longbool;
    g_Msg1: TMsg;
    g_VideoAken: TPOINT;
    g_6petus: array[1..2000] of byte;

const WM_CAP_START = WM_USER;
      {WM_CAP_STOP = WM_CAP_START+68;}
      WM_CAP_DRIVER_CONNECT = WM_CAP_START+10;
      WM_CAP_DRIVER_DISCONNECT = WM_CAP_START+11;
      {WM_CAP_SAVEDIB = WM_CAP_START+25;}
      {WM_CAP_GRAB_FRAME = WM_CAP_START+60;}
      {WM_CAP_SEQUENCE = WM_CAP_START+62;}
      {WM_CAP_SEQUENCE_NOFILE = WM_CAP_START+63;}
      {WM_CAP_SET_SEQUENCE_SETUP = WM_CAP_START+64;}
      {WM_CAP_GET_SEQUENCE_SETUP = WM_CAP_START+65;}
      {WM_CAP_FILE_SET_CAPTURE_FILEA = WM_CAP_START+20;}
      WM_CAP_SET_PREVIEW = WM_CAP_START+50;
      WM_CAP_SET_OVERLAY = WM_CAP_START+51;
      WM_CAP_SET_PREVIEWRATE = WM_CAP_START+52;
      WM_CAP_SET_SCALE = WM_CAP_START+53;
      WM_CAP_SET_SCROLL = WM_CAP_START+55;
      {WM_CAP_EDIT_COPY = WM_CAP_START+30;}
      {WM_CAP_DLG_VIDEOFORMAT = WM_CAP_START+41;}
      {WM_CAP_DLG_VIDEOSOURCE = WM_CAP_START+42;}
      {WM_CAP_DLG_VIDEODISPLAY = WM_CAP_START+43;}
      WM_CAP_GET_VIDEOFORMAT = WM_CAP_START+44;
      WM_CAP_SET_VIDEOFORMAT = WM_CAP_START+45;
      WM_CAP_SET_CALLBACK_FRAME = WM_CAP_START+5;
      {WM_CAP_SET_CALLBACK_VIDEOSTREAM = WM_CAP_START+6;}

{$R fabry.res}

procedure piip; begin messagebeep($FFFF); end;

function capCreateCaptureWindowA(lpszWindowName: pChar; dwStyle: longint; x, y, dx, dy: integer;
                                 ParentWin: HWND; nId: integer): HWND; stdcall;
                                 external 'AVICAP32.DLL';
{function GPIOIsInitOK: boolean; stdcall; external 'BTVID_32';}
{function GetGPCLKMODE: integer; stdcall; external 'BTVID_32';}  {v6i word ?????}
{procedure SetGPCLKMODE(mode: integer); stdcall; external 'BTVID_32';}
{function Bt848GetVideoSource(p: pinteger): integer; stdcall; external 'BTVID_32';}
function Bt848SetVideoSource(i: integer): integer; stdcall; external 'BTVID_32';
{function Bt848GetBrightness(p: pinteger): integer; stdcall; external 'BTVID_32';}
function Bt848SetBrightness(i: integer): integer; stdcall; external 'BTVID_32';
{function Bt848GetContrast(p: pinteger): integer; stdcall; external 'BTVID_32';}
function Bt848SetContrast(i: integer): integer; stdcall; external 'BTVID_32';
{function Bt848GetHue(p: pinteger): integer; stdcall; external 'BTVID_32';}
{function Bt848SetHue(i: integer): integer; stdcall; external 'BTVID_32';}
{function Bt848GetSaturation(p: pinteger): integer; stdcall; external 'BTVID_32';}
function Bt848SetSaturation(i: integer): integer; stdcall; external 'BTVID_32';
{function Bt848GetStatus: longint; stdcall; external 'BTVID_32';}
{function Bt848GetVideoFormat(p: pinteger): integer; stdcall; external 'BTVID_32';}
function Bt848SetVideoFormat(i: integer): integer; stdcall; external 'BTVID_32';  {PAL-B on 3}
{function GetInterlaceMode: integer; stdcall; external 'BTVID_32';}
{function SetInterlaceMode(i: integer): integer; stdcall; external 'BTVID_32';} {0, 1, 2}
{procedure Bt848Pause; stdcall; external 'BTVID_32';}
{procedure Bt848Continue; stdcall; external 'BTVID_32';}
{function Bt848GetOverlayFlags: longint; stdcall; external 'BTVID_32';}  {1, 2}
```

```
{function Bt848ImageLoad(HI: word; Wnd: hWND; BM: LPCSTR): integer; stdcall; external
'BTVID_32';}
{function Bt848UnLoadImage: integer; stdcall; external 'BTVID_32';}
{function GetVideoInfo(p: pointer): integer; stdcall; external 'Bt848Dlg';}
{function GetVideoInfo(p: pointer): integer; stdcall; external 'Bt848Dlg';}
{function ReadDecoder(reg: word; value: pointer): integer; stdcall; external 'BTVID_32';}
function WriteDecoder(reg: word; value: byte): integer; stdcall; external 'BTVID_32';
{function SetVideoInfo(p: pointer): integer; stdcall; external 'BT848DLG';}
function Bt848SetImage(p1: pRect; i: integer): integer; stdcall; external 'BTVID_32';
function Bt848GetImageFormat(p: pInteger): integer; stdcall; external 'BTVID_32';
procedure Bt848GetImageSize(p: pRect); stdcall; external 'BTVID_32';

function OpenFileName(b: byte; var nimi: string): LONGBOOL;
var f: TOpenFileName;
    buf1, buf2: array[0..255] of char;
begin
OpenFileName:=FALSE;
FillChar(f,sizeOf(f),#0);
FillChar(buf1,SizeOf(buf1),#0);
FillChar(buf2,SizeOf(buf2),#0);
buf1:='dat file'+#0+'*.dat'+#0+'Any File'+#0+'*.*'+#0+#0;
f.lStructSize:=sizeof(f);
f.lpstrFilter:=@buf1;
f.nMaxFile:=sizeof(buf2);
f.lpstrFile:=@buf2;
f.lpstrInitialDir:=@g_DataPath;
{fp:}
{if b=1 then OpenFileName:=GetOpenFileName(@f) else OpenFileName:=GetSaveFileName(@f);}
{Delphi:}
if b=1 then OpenFileName:=GetOpenFileName(f) else OpenFileName:=GetSaveFileName(f);
nimi:=StrPas(@buf2)+#0;
end;  {OpenFileName}

function OnNimi(nimi: pChar): LONGBOOL;  {!!!!!}
var i, j: integer;
    buf: array[0..50] of char;
begin
StrLCopy(buf,nimi,50);
OnNimi:=FALSE;
i:=StrLen(buf);
if i=0 then exit;
for j:=1 to i-1 do if (not(buf[j] in ['A'..'Z','a'..'z','0'..'9','_','-',' '])) then exit;
OnNimi:=TRUE;
end;

function StrToLong(var s: string; var l: longint): LONGBOOL;  {leiab stringist t@isarvu}
var i, j: integer;
begin
StrToLong:=FALSE;
i:=1;
while (not (s[i] in ['0'..'9'])) do begin s[i]:=#32; inc(i); end;
j:=i;
while (s[i] in ['0'..'9']) do inc(i);
s:=copy(s,j,i-j);
val(s,l,i);
if i=0 then StrToLong:=TRUE;
end;

function Salvesta(nimi: pChar; p: pAndmed): LONGBOOL;
var f: text;
    i: integer;
begin
Salvesta:=FALSE;
Assign(f,StrPas(nimi));
{$I-} ReWrite(f); {$I+}
if IOResult<>0 then exit;
with p^
do begin
   writeln(f,'Fabry95 andmefail');
   writeln(f,aeg);
   writeln(f,'Keskmistatud kaadreid: ',pilte);
   writeln(f,'Ridu igas kaadris: ',ridu);
   writeln(f,'Piksleid igas reas: ',punkte);
   writeln(f,'Kommentaar: ',komm);
   writeln(f{,'Polynoomi kordajad:'});
   writeln(f{,'K0=',k0:5});
   writeln(f{,'K1=',k1:5});
```

```
   writeln(f{,'K2=',k2:5});
   writeln(f{,'K3=',k3:5});
   writeln(f{,'K4=',k4:5});
   writeln(f{,'K5=',k5:5});
   writeln(f{,'K6=',k6:5});
   writeln(f{,'K7=',k7:5});
   writeln(f{,'K8=',k8:5});
   for i:=1 to 4 do writeln(f);
   for i:=1 to punkte do writeln(f,g_m1[i],' ',g_m2[i]{,' ',g_m3[i]});
   end;  {with}
Close(f);
end;  {Salvesta}

function LoeSisse(nimi: pChar; var viga: integer): LONGBOOL;
var f: text;
    i: integer;
    s: string;
    bl: LONGBOOL;
label 1;
begin
LoeSisse:=FALSE;
Assign(f,nimi);
s:=ExtractFileName(StrPas(nimi));  {nimi koos laiendiga}
StrPCopy(g_AvaHeader.nimi,s);
IOResult;
{$I-} Reset(f);
if IOResult<>0 then begin viga:=-1; {$I+} exit; end;
with g_AvaHeader
do begin
   viga:=0;
   readln(f,s);  {ID}
   if Pos('andmefail',s)=0 then goto 1;
   inc(viga);
   readln(f,aeg);
   inc(viga);
   readln(f,s);  {pilte}
   bl:=StrToLong(s,pilte);
   if ((not bl) or (pilte<1) or (pilte>1000)) then goto 1;
   inc(viga);
   readln(f,s);  {ridu}
   bl:=StrToLong(s,ridu);
   if ((not bl) or (ridu<1) or (ridu>1000)) then goto 1;
   inc(viga);
   readln(f,s);  {punkte}
   bl:=StrToLong(s,punkte);
   if ((not bl) or (punkte<100) or (punkte>1000)) then goto 1;
   inc(viga);
   readln(f,s);  {komm}
   s:=copy(s,13,255);
   StrPCopy(komm,s);
   inc(viga); readln(f,s); if IOResult<>0 then goto 1;
   inc(viga); readln(f,s); {val(copy(s,4,20),k0,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k1,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k2,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k3,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k4,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k5,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k6,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k7,i); if (i<>0) then goto 1;}
   inc(viga); readln(f,s); {val(copy(s,4,20),k8,i); if (i<>0) then goto 1;}
   for i:=1 to 4 do begin readln(f); if IOResult<>0 then goto 1; end;
   max1:=1; g_m1[max1]:=0;
   max2:=1; g_m2[max2]:=0;
   max3:=1; g_m3[max3]:=0;
   for i:=1 to punkte do begin
                         inc(viga);
                         readln(f,g_m1[i],g_m2[i]{,g_m3[i]});
                         if IOResult<>0 then goto 1;
                         if g_m1[i]>g_m1[max1] then max1:=i;
                         if g_m2[i]>g_m2[max2] then max2:=i;
                         {if g_m3[i]>g_m3[max3] then max3:=i;}
                         end;
   end;  {with}
Close(f); {$I+}
if IOResult<>0 then begin viga:=-2; goto 1; end;
LoeSisse:=TRUE;
exit;
```

```
1: Close(f); {$I+}
end;  {Loe Sisse}

function CallBack2(h: HWND; p: PVIDEOHDR): DWORD; stdcall;
var p1, p2: pointer;
    l1, l2: longint;
begin
CallBack2:=1;
if g_Sehkendan then exit;
l1:=(g_VideoAken.y-g_Rida2)*2*g_VideoAken.x;
l2:=(g_Rida2-g_Rida1)*2*g_VideoAken.x;
p1:=p^.lpData;
p2:=g_pBuf1;
asm
   pushf
   push     eax
   push     ebx
   push     ecx
   push     edx
   mov      ebx, p1
   add      ebx, l1  {300*2*768 4608}
   mov      edx, p2
   mov      ecx, l2  {50*768}
@l1:
   mov      ax, word ptr [ebx]
   mov      word ptr [edx], ax
   add      ebx, 2
   add      edx, 2
   loop     @l1
   pop      edx
   pop      ecx
   pop      ebx
   pop      eax
   popf
end;
g_Sehkendan:=TRUE;
PostMessage(g_hwMain,WM_COMMAND,802,0);
end;

function KoheProc(DlgWin: hwnd; Msg, wParam, lParam: longint): longint; stdcall;
begin
KoheProc:=1;
case Msg
of WM_INITDIALOG: begin
                  SetWindowPos(DlgWin,HWND_TOP,(g_MaxX-150) shr 1,
                               (g_MaxY-50) shr 1,0,0,SWP_NOSIZE);
                  exit;
                  end;  {WM_INITDIALOG}
   else begin KoheProc:=0; end;
end;  {case message}
end;  {KoheProc}

function TeineProc(hw2: hwnd; Msg, wParam, lParam: longint): longint; stdcall;
begin
TeineProc:=0;
TeineProc:=DefWindowProc(hw2,Msg,WParam,LParam);
end;  {TeineProc}

function MainProc(hw1: hwnd; Msg, wParam, lParam: longint): longint; stdcall;
var i1, i2, i3: integer;
    w1: word;
    buf1, buf2: array[0..255] of char;
    s1: string;
    b1: boolean;
    l1, l2: longint;
    e1: extended;
    r1: real;
    p1, p2: pointer;
    PS1: TPaintStruct;
    TR1: TRect;
    CP1: TCAPTUREPARMS;
    VH1: VIDEOHDR;
begin
MainProc:=0;
case Msg
of WM_PAINT: begin
             BeginPaint(g_hwMain,PS1);
```

```
case g_Olukord
of 0: begin
      end;
   1: begin
      end;
   2: begin
      SelectObject(g_DCMain,g_hf1);  {ANSI_VAR_FONT}
      SelectObject(g_DCMain,g_hp1);  {j@me must}
      MoveToEx(g_DCMain,g_x1,g_y1,nil); LineTo(g_DCMain,g_x1+g_dx1,g_y1);
      MoveToEx(g_DCMain,g_x2,g_y2,nil); LineTo(g_DCMain,g_x2+g_dx2,g_y2);
      SetTextAlign(g_DCMain,TA_RIGHT+TA_TOP);
      SelectObject(g_DCMain,g_hp0);  {peenike must}
      MoveToEx(g_DCMain,g_x1,g_y1,nil); LineTo(g_DCMain,g_x1,g_y1-g_dy1);
      MoveToEx(g_DCMain,g_x1+g_dx1,g_y1,nil);
      LineTo(g_DCMain,g_x1+g_dx1,g_y1-g_dy1);
      with g_AvaHeader
      do begin
         for i1:=1 to 10
         do begin
            MoveToEx(g_DCMain,g_x1,round(g_y1-i1*g_dy11),nil);
            LineTo(g_DCMain,g_x1+5,round(g_y1-i1*g_dy11));
            MoveToEx(g_DCMain,g_x1+g_dx1,round(g_y1-i1*g_dy11),nil);
            LineTo(g_DCMain,g_x1-5+g_dx1,round(g_y1-i1*g_dy11));
            str(10*i1,s1);
            if i1=10 then s1:='%';
            TextOut(g_DCMain,g_x1-2,g_y1-5-round(i1*g_dy11),@s1[1],length(s1));
            end;
         SelectObject(g_DCMain,g_hp2);  {peenike roheline}
         MoveToEx(g_DCMain,g_x2,g_y2,nil); LineTo(g_DCMain,g_x2,g_y2-g_dy2);
         for i1:=1 to 10  {?????}
         do begin
            MoveToEx(g_DCMain,g_x2,g_y2-round(i1*g_dy21),nil);
            LineTo(g_DCMain,g_x2+5,g_y2-round(i1*g_dy21));
            str(10*i1,s1);
            if i1=10 then s1:='%';
            TextOut(g_DCMain,g_x2-2,g_y2-5-round(i1*g_dy21),@s1[1],length(s1));
            end;
         MoveToEx(g_DCMain,g_x2,g_y2-round(ky2*g_m2[1]),nil);
         for i1:=2 to g_AvaHeader.punkte
         do LineTo(g_DCMain,g_x2+round(i1*dx21),g_y2-round(ky2*g_m2[i1]));
         MoveToEx(g_DCMain,g_x2+20,g_y2-g_dy2-10,nil);
         LineTo(g_DCMain,g_x2+40,g_y2-g_dy2-10);
         SetTextAlign(g_DCMain,TA_LEFT+TA_TOP);
         SelectObject(g_DCMain,g_hp4);  {peenike punane}
         MoveToEx(g_DCMain,g_x2+g_dx2,g_y2,nil);
         LineTo(g_DCMain,g_x2+g_dx2,g_y2-g_dy2);
         for i1:=1 to 10  {?????}
         do begin
            MoveToEx(g_DCMain,g_x2+g_dx2,g_y2-round(i1*g_dy21),nil);
            LineTo(g_DCMain,g_x2-5+g_dx2,g_y2-round(i1*g_dy21));
            str(10*i1,s1);
            if i1=10 then s1:='%';
            TextOut(g_DCMain,g_x2+2+g_dx2,g_y2-5-round(i1*g_dy21),@s1[1],
                    length(s1));
            end;
         s1:='pimesignaal';
         TextOut(g_DCMain,g_x2+42,g_y2-g_dy2-17,@s1[1],length(s1));
         SelectObject(g_DCMain,g_hp0);  {peenike must}
         MoveToEx(g_DCMain,g_x1+20,g_y1-g_dy1-10,nil);
         LineTo(g_DCMain,g_x1+40,g_y1-g_dy1-10);
         s1:='interferogramm';
         TextOut(g_DCMain,g_x1+42,g_y1-g_dy1-17,@s1[1],length(s1));
         SetTextAlign(g_DCMain,TA_CENTER+TA_TOP);
         for i1:=0 to 15
         do begin
            MoveToEx(g_DCMain,g_x1+round(i1*dx12),g_y1,nil);
            LineTo(g_DCMain,g_x1+round(i1*dx12),g_y1+5);
            MoveToEx(g_DCMain,g_x2+round(i1*dx22),g_y2,nil);
            LineTo(g_DCMain,g_x2+round(i1*dx22),g_y2+5);
            str(50*i1,s1);
            TextOut(g_DCMain,g_x1+round(i1*dx12),g_y1+5,@s1[1],length(s1));
            TextOut(g_DCMain,g_x2+round(i1*dx22),g_y2+5,@s1[1],length(s1));
            end;
         MoveToEx(g_DCMain,g_x1,g_y1-round(ky1*g_m1[1]),nil);  {interferogramm}
         for i1:=2 to g_AvaHeader.punkte
         do LineTo(g_DCMain,g_x1+round(i1*dx11),g_y1-round(ky1*g_m1[i1]));
         SelectObject(g_DCMain,g_hf0);
```

```
                  SetTextAlign(g_DCMain,TA_LEFT+TA_TOP);
                  with TR1 do begin
                              top:=g_y4-g_dy4;
                              bottom:=g_y4;
                              left:=g_x4;
                              right:=g_MaxX-20;
                              end;
                  buf1:='Fail: ';
                  StrCat(buf1,nimi);
                  i1:=DrawTextEx(g_DCMain,buf1,-1,TR1,DT_WORDBREAK,nil);
                  with TR1 do top:=top+i1;
                  buf1:='Mõõdetud: ';
                  StrCat(buf1,aeg);
                  i1:=DrawTextEx(g_DCMain,buf1,-1,TR1,DT_WORDBREAK,nil);
                  with TR1 do top:=top+i1;
                  str(pilte,s1);
                  s1:='Keskmistatud '+s1+' kaadrit';
                  StrPCopy(buf1,s1);
                  i1:=DrawTextEx(g_DCMain,buf1,-1,TR1,DT_WORDBREAK,nil);
                  with TR1 do top:=top+i1;
                  buf1:='Kommentaar: ';
                  StrCat(buf1,komm);
                  i1:=DrawTextEx(g_DCMain,buf1,-1,TR1,DT_WORDBREAK,nil);  {tagastab k6rguse}
                  end;  {with}
               end;
            3: begin  {justeerimine}
               SelectObject(g_DCMain,g_hp0);  {peenike must}
               MoveToEx(g_DCMain,0,0,nil);
               LineTo(g_DCMain,g_MaxX,0);
               end;
         4, 5: begin  {p6hiline ja myra m66tmine}
               SelectObject(g_DCMain,g_hp1);  {j@me must}
               MoveToEx(g_DCMain,g_x8-2,g_y5-g_dy5,nil);
               LineTo(g_DCMain,g_x8-2,g_y5+2);
               LineTo(g_DCMain,g_x8+g_dx8+2,g_y5+2);
               LineTo(g_DCMain,g_x8+g_dx8+2,g_y5-g_dy5);
               MoveToEx(g_DCMain,g_x8-2,g_y6-g_dy6,nil);
               LineTo(g_DCMain,g_x8-2,g_y6+2);
               LineTo(g_DCMain,g_x8+g_dx8+2,g_y6+2);
               LineTo(g_DCMain,g_x8+g_dx8+2,g_y6-g_dy6);
               SelectObject(g_DCMain,g_hp0);  {peenike must}
               MoveToEx(g_DCMain,0,0,nil);
               LineTo(g_DCMain,g_MaxX,0);
               w1:=g_Buf1[1] shr 8;  {!!!!!}
               MoveToEx(g_DCMain,g_x8,g_y5-w1,nil);  {ylemine graafik}
               for i1:=2 to g_T66Header.punkte
               do begin
                  w1:=g_Buf1[i1] shr 8;
                  LineTo(g_DCMain,g_x8+round(i1*g_T66Header.dx51),g_y5-w1);
                  end;
               if g_Olukord=4  {panna yhte massiivi, p@rast kopeerida !!!!!}
               then begin  {myra m66tmine}
                    l2:=g_m2[1] shr g_T66Header.nih1;
                    MoveToEx(g_DCMain,g_x8,g_y6-l2,nil);
                    for i1:=2 to g_T66Header.punkte
                    do begin
                       l2:=g_m2[i1] shr g_T66Header.nih1;
                       LineTo(g_DCMain,g_x8+round(i1*g_T66Header.dx51),g_y6-l2);
                       end;
                    end
               else begin  {p6hiline m66tmine}
                    l2:=g_m1[1] shr g_T66Header.nih1;
                    MoveToEx(g_DCMain,g_x8,g_y6-l2,nil);
                    for i1:=2 to g_T66Header.punkte
                    do begin
                       l2:=g_m1[i1] shr g_T66Header.nih1;
                       LineTo(g_DCMain,round(g_x8+i1*g_T66Header.dx51),g_y6-l2);
                       end;
                    end;
               end;
         end;  {case}
         EndPaint(g_hwMain,PS1);
         exit;
         end;  {WM_PAINT}
WM_ERASEBKGND: begin
               MainProc:=1;
               with g_TR1 do begin top:=0; bottom:=g_MaxY; left:=0; right:=g_MaxX; end;
```

```
                case g_OluKord
                of 0: begin g_hb0:=g_hb1; end;
                   1: begin g_hb0:=g_hb1; end;  {tyhi aken}
                   2: begin g_hb0:=g_hb2; end;  {avatud fail}
                   3: begin g_hb0:=g_hb3; end;  {h@@lestamine}
                   4, 5: begin g_hb0:=g_hb3; end;  {m66tmine}
                end;
                FillRect(g_DCMain,g_TR1,g_hb0);
                exit;
                end;  {WM_ERASEBKGND}

WM_SIZE: begin
         MoveWindow(g_hwStatus,0,0,0,0,TRUE);
         exit;
         end;
WM_HSCROLL: begin  {heleduse ja kontrasti trackbarid}
            if LoWord(wParam)=8 then exit;
            if LoWord(wParam)=4 then exit;
            if lParam<>g_hwm[3] then exit;
            if LoWord(wParam)=5
            then l1:=hiword(wParam)
            else l1:=SendMessage(g_hwm[3],TBM_GETPOS,0,0);
            if g_Heledus<>l1
            then begin
                 g_Heledus:=l1;
                 Bt848SetBrightness(g_Heledus);
                 end;
            exit;
            end;
WM_COMMAND: begin
            case wparam
            of 101: begin  {Open}
                    if OpenFileName(1,g_s1)
                    then begin
                         if LoeSisse(@g_s1[1],g_i1)
                         then begin
                              PostMessage(g_hwMain,WM_COMMAND,803,0);
                              exit;
                              end
                         else begin
                              str(g_i1,s1);
                              s1:='Fail on vigane, kood '+s1+#0;
                              i1:=MessageBox(g_hwMain,@s1[1],'Viga:',
                                             MB_OK+MB_ICONEXCLAMATION);
                              exit;
                              end;
                         end
                    else exit;
                    end;
               102: begin  {Save As...}
                    if OpenFileName(2,g_s1)
                    then begin
                         s1:=ExtractFileName(g_s1);
                         StrPCopy(g_AvaHeader.nimi,s1);
                         Salvesta(StrPCopy(@buf1,g_s1),@g_AvaHeader);
                         StrPCopy(buf1,'Fabry95: ');
                         StrCat(buf1,g_AvaHeader.nimi);
                         SetWindowText(g_hwMain,buf1);
                         end;
                    exit;
                    end;
               103: begin  {Close}
                    case g_Olukord
                    of 2: begin  {fail on avatud}
                          EnableMenuItem(g_hm1,101,MF_ENABLED);  {Open}
                          EnableMenuItem(g_hm1,102,MF_GRAYED);  {Save As...}
                          EnableMenuItem(g_hm1,103,MF_GRAYED);  {Close}
                          EnableMenuItem(g_hm1,104,MF_GRAYED);  {Print}
                          g_Olukord:=1;
                          SetWindowText(g_hwMain,'Fabry95');
                          InvalidateRect(g_hwMain,nil,TRUE);
                          PostMessage(g_hwMain,WM_PAINT,0,0);
                          end;
                       3, 4, 5: begin  {h@@lestamine, m66tmine}
                                PostMessage(g_hwMain,WM_COMMAND,202,0);
                                end;
                    end;  {case}
```

```
     exit;
     end;
104: begin  {Print}
     exit;
     end;
105: begin  {Exit}
     PostMessage(g_hwMain,WM_CLOSE,0,0);
     exit;
     end;
201: begin  {M66da}
     SetWindowText(g_hwStatus,'');
     EnableMenuItem(g_hm1,101,MF_GRAYED);  {Open}
     EnableMenuItem(g_hm1,102,MF_GRAYED);  {Save As...}
     EnableMenuItem(g_hm1,103,MF_ENABLED);  {Close}
     EnableMenuItem(g_hm1,104,MF_GRAYED);  {Print}
     EnableMenuItem(g_hm1,201,MF_GRAYED);  {M66da}
     EnableMenuItem(g_hm1,202,MF_ENABLED);  {Katkesta}
     if g_Olukord=2
     then begin
          g_Olukord:=1;
          SendMessage(g_hwMain,WM_ERASEBKGND,0,0);
          end;
     g_hwKohe:=CreateDialog(hInstance,'KOHE',g_hwMain,@KoheProc);
     SetDlgItemText(g_hwKohe,101,'Aparatuuri initsialiseerimine...');
     g_hwClip:=CreateWindow('teine','',WS_CHILD+WS_VISIBLE+WS_BORDER,
                            g_x8,g_y8,g_dx8,g_dy8,g_hwMain,0,HInstance,nil);
     g_hwCap:=capCreateCaptureWindowA('blaa',WS_CHILD+WS_VISIBLE,
                                      0,0-g_Rida1,g_VideoAken.x,
                                      g_VideoAken.y,g_hwClip,0);
     i1:=SendMessage(g_hwCap,WM_CAP_DRIVER_CONNECT,0,0);
     i1:=SendMessage(g_hwCap,WM_CAP_SET_CALLBACK_FRAME,0,longint(@CallBack2));
     SendMessage(g_hwCap,WM_CAP_SET_OVERLAY,1,0);
     SendMessage(g_hwCap,WM_CAP_SET_SCALE,0,0);
     Bt848SetVideoSource(3);
     Bt848SetVideoFormat(3);
     Bt848SetSaturation(0);
     Bt848SetContrast(255);  {!!!!!}
     Bt848SetBrightness(128);
     i1:=SendMessage(g_hwCap,WM_CAP_GET_VIDEOFORMAT,
                     sizeof(bmi1),longint(@bmi1));
     with bmi1.bmiHeader
     do begin
        biWidth:=768;
        biHeight:=576;
        biSize:=SizeOf(TBitmapInfoHeader);
        biCompression:=BI_RGB;
        biXPelsPerMeter:=0;
        biYPelsPerMeter:=0;
        biClrUsed:=0;
        biClrImportant:=0;
        biPlanes:=1;
        biBitCount:=16;
        biSizeImage:={WidthBytes(biWidth*biBitCount)*biHeight*biPlanes}885000;
        end;  {with}
     i1:=SendMessage(g_hwCap,WM_CAP_SET_VIDEOFORMAT,
                     sizeof(bmi1),LONGINT(@bmi1));
     Bt848GetImageSize(@TR1);
     Bt848SetImage(@TR1,3);       {3}
     Bt848GetImageFormat(@i1);
     WriteDecoder($0048,128);  {Coring}
     DestroyWindow(g_hwKohe);
     SetWindowText(g_hwStatus,'Justeeri pilt ja täida lahtrid');
     {}
     g_hwm[1]:=CreateWindow('BUTTON','Alustan mõõtmist',
                            BS_PUSHBUTTON+WS_CHILD+WS_VISIBLE,
                            230,g_y8+g_dy8+180,200,25,g_hwMain,801,0,nil);
     g_hwm[2]:=CreateWindow('BUTTON','Loobun mõõtmisest',WS_CHILD+WS_VISIBLE,
                            440,g_y8+g_dy8+180,200,25,g_hwMain,202,0,nil);
     g_hwm[3]:=CreateWindow('MSCTLS_TRACKBAR32','',WS_CHILD+WS_VISIBLE,
                            210,g_y8+g_dy8+20,150,30,g_hwMain,803,0,nil);
     SendMessage(g_hwm[3],TBM_SETRANGE,1,$00ff0000);
     SendMessage(g_hwm[3],TBM_SETPOS,1,128);
     g_hwm[5]:=CreateWindow('STATIC','Heledus',WS_CHILD+WS_VISIBLE+SS_RIGHT,
                            100,g_y8+g_dy8+20,100,20,g_hwMain,805,0,nil);
     g_hwm[7]:=CreateWindow('STATIC','',WS_CHILD+WS_VISIBLE,
                            25,10,g_MaxX-50,g_y8-26,g_hwMain,807,0,nil);
     SetWindowText(g_hwm[7],@g_6petus);
```

```
     g_hwm[4]:=CreateWindow('EDIT','blaa-blaa',
                            WS_CHILD+WS_BORDER+WS_VISIBLE+ES_AUTOHSCROLL,
                            230,g_y8+g_dy8+140,500,20,g_hwMain,809,0,nil);
     g_hwm[6]:=CreateWindow('STATIC','Kommentaar:',
                            WS_CHILD+SS_RIGHT+WS_VISIBLE,
                            100,g_y8+g_dy8+140,120,20,g_hwMain,810,0,nil);
     g_hwm[8]:=CreateWindow('EDIT','fabry',
                            WS_CHILD+WS_BORDER+ES_AUTOHSCROLL+WS_VISIBLE,
                            230,g_y8+g_dy8+100,100,20,g_hwMain,808,0,nil);
     g_hwm[9]:=CreateWindow('EDIT','10',WS_CHILD+WS_BORDER+WS_VISIBLE,
                            230,g_y8+g_dy8+60,40,20,g_hwMain,809,0,nil);
     g_hwm[10]:=CreateWindow('STATIC','Faili nimi:',
                             WS_CHILD+SS_RIGHT+WS_VISIBLE,
                             100,g_y8+g_dy8+100,120,20,g_hwMain,810,0,nil);
     g_hwm[11]:=CreateWindow('STATIC','Keskmistada üle',
                             WS_CHILD+SS_RIGHT+WS_VISIBLE,
                             100,g_y8+g_dy8+60,120,20,g_hwMain,811,0,nil);
     g_hwm[12]:=CreateWindow('STATIC',@g_DefaultHeader.laiend,
                             WS_CHILD+WS_VISIBLE,
                             332,g_y8+g_dy8+100,50,20,g_hwMain,812,0,nil);
     g_hwm[13]:=CreateWindow('STATIC','kaadri',WS_CHILD+WS_VISIBLE,
                             280,g_y8+g_dy8+60,100,20,g_hwMain,813,0,nil);
     g_Olukord:=3;
     InvalidateRect(g_hwMain,nil,TRUE);
     PostMessage(g_hwMain,WM_PAINT,0,0);
     {}
     exit;
     end;  {Uus M66tmine}
202: begin  {Katkesta}
     i1:=SendMessage(g_hwCap,WM_CAP_SET_CALLBACK_FRAME,0,0);
     SendMessage(g_hwCap,WM_CAP_DRIVER_DISCONNECT,0,0);
     DestroyWindow(g_hwCap);
     DestroyWindow(g_hwClip);
     for i2:=1 to 13 do DestroyWindow(g_hwm[i2]);  {i1 ei sobi}
     if g_Olukord=6
     then begin  {korraline m66tmise l6pp}
          g_Olukord:=2;
          Move(g_T66Header,g_AvaHeader,SizeOf(andmed));
          StrCat(g_AvaHeader.nimi,g_AvaHeader.laiend);
          PostMessage(g_hwMain,WM_COMMAND,803,0);
          exit;
          end
     else begin
          EnableMenuItem(g_hm1,101,MF_ENABLED);  {Open}
          EnableMenuItem(g_hm1,103,MF_GRAYED);  {Close}
          EnableMenuItem(g_hm1,201,MF_ENABLED);  {M66da}
          EnableMenuItem(g_hm1,202,MF_GRAYED);  {Katkesta}
          g_Olukord:=1;  {tyhi taust}
          SetWindowText(g_hwMain,'Fabry95');
          SetWindowText(g_hwStatus,'Mõõtmiseks vali menüüst "Mõõda"');
          InValidateRect(g_hwMain,nil,TRUE);
          PostMessage(g_hwMain,WM_PAINT,0,0);
          exit;
          end;
     end;  {GoTo...}
301: begin  {About}
     i1:=MessageBox(g_hwMain,'Programm spektrijoone ülipeenstruktuuri'+
                    #13+'määramiseks Fabry-Perot'' etaloni abil',
                    'Programmist:',MB_OK+MB_ICONINFORMATION);
     exit;
     end;
801: begin  {M66da}
     SetWindowText(g_hwStatus,'Loen initsialiseerimisfaili...');
     if (not LoeSisse(@g_MyraFail,g_i1))
     then begin
          str(g_i1,s1);
          s1:='Fail on vigane, kood '+s1+#0;
          i1:=MessageBox(g_hwMain,@s1[1],'Viga:',MB_OK+MB_ICONEXCLAMATION);
          g_Olukord:=1;
          PostMessage(g_hwMain,WM_COMMAND,202,0);
          exit;
          end;
     Move(g_AvaHeader,g_T66Header,SizeOf(andmed));
     Move(g_DefaultHeader.laiend,g_T66Header.laiend,20);
     g_T66Header.ridu:=g_DefaultHeader.ridu;
     i1:=GetWindowText(g_hwm[8],@buf1,50);  {faili nimi}
     if ((i1=0) or (not OnNimi(@buf1)))
```

```
     then begin
          i1:=MessageBox(g_hwMain,'Faili nimi on vigane!','Viga:',
                         MB_OK+MB_ICONEXCLAMATION);
          exit;
          end
     else begin
          Move(buf1,g_T66Header.nimi,i1+1);  {!!!!!}
          end;
     i1:=GetWindowText(g_hwm[9],@buf1,20);  {kordus}
     g_T66Header.pilte:=StrToIntDef(buf1,10);
     i1:=GetWindowText(g_hwm[4],@g_T66Header.komm,255);  {kommentaar}
     SetWindowText(g_hwStatus,'');
     for i1:=1 to 13 do DestroyWindow(g_hwm[i1]);
     Bt848SetContrast(150);  {!!!!!}
     Bt848SetBrightness(128);
     FillMemory(@g_m1,3200,0);
     FillMemory(@g_m2,3200,0);
     g_T66Header.max2:=0;  {myra oma}
     g_T66Header.nih1:=0;
     {g_T66Header.punkte:=768;}  {!!!!!}
     SendMessage(g_hwCap,WM_CAP_SET_PREVIEWRATE,0,0);
     SendMessage(g_hwCap,WM_CAP_SET_PREVIEW,1,0);
     g_Punkt1.x:=0;
     g_Punkt1.y:=g_Rida1;
     i1:=SendMessage(g_hwCap,WM_CAP_SET_SCROLL,1,longint(@g_Punkt1));
     SetWindowPos(g_hwCap,HWND_TOP,0,0,g_VideoAken.x,g_Rida1,SWP_SHOWWINDOW);
     i1:=MessageBox(g_hwMain,'Sulge katik ja pressi OK!','',MB_OK);
     SendMessage(g_hwCap,WM_CAP_SET_PREVIEWRATE,500,0);
     SetWindowText(g_hwStatus,'Mõõdan.
                   Katkestamiseks vali "Mõõda"-menüüst "Katkesta"');
     g_Olukord:=4;  {myra m66tmine}
     g_Punkt:=1;
     g_Sehkendan:=FALSE;
     InvalidateRect(g_hwMain,nil,TRUE);
     PostMessage(g_hwMain,WM_PAINT,0,0);
     exit;
     end;
802: begin  {yks kaader m66detud}
     {}
     if (not g_Olukord in [4,5]) then exit;  {?????}
     p1:=g_pBuf1;  {src}
     p2:=@g_Buf1;  {dest}
     l1:=0;
     for i3:=0 to (g_Rida2-g_Rida1)  {see i konflikteerub WM_PAINT omaga !!!!!}
     do begin
        asm
        pushf
        push     eax
        push     ebx
        push     ecx
        push     edx
        mov      ebx, p1
        add      ebx, 0
        add      ebx, l1
        mov      edx, p2
        mov      ecx, 768
   @l1:
        mov      ax, word ptr [ebx]
        mov      word ptr [edx], ax
        add      ebx, 2
        add      edx, 2
        loop     @l1
        pop      edx
        pop      ecx
        pop      ebx
        pop      eax
        popf
        end;
        inc(l1,2*768);
        with g_T66Header
        do begin
           if g_Olukord=4  {myra m66tmine}
           then begin
                for i2:=1 to 768
                do begin
                   g_m2[i2]:=g_m2[i2]+g_Buf1[i2];
                   if g_m2[i2]>g_m2[max2] then max2:=i2;
```

```
                    end;
                 while ((g_m2[max2] shr nih1)>150) do inc(nih1);
                 end
           else begin  {p6hiline m66tmine}
                 for i2:=1 to 768
                 do begin
                    g_m1[i2]:=g_m1[i2]+g_Buf1[i2];
                    if g_m1[i2]>g_m1[max1] then max1:=i2;
                    end;
                 while ((g_m1[max1] shr nih1)>150) do inc(nih1);
              end;
           end;  {with}
        end;  {i3}
     {}
     with TR1 do begin
                 top:=g_y5-g_dy5;
                 bottom:=g_y5;
                 left:=g_x8;
                 right:=g_x8+g_dx8;
                 end;
     InValidateRect(g_hwMain,@TR1,TRUE);
     with TR1 do begin top:=g_y6-g_dy6; bottom:=g_y6; end;
     InValidateRect(g_hwMain,@TR1,TRUE);
     PostMessage(g_hwMain,WM_PAINT,0,0);
     if g_Punkt<g_T66Header.pilte
     then begin
          inc(g_Punkt);
          g_Sehkendan:=FALSE;
          exit;
          end
     else begin  {viimane punkt}
          if g_Olukord=4
          then begin  {myra m66tmine}
               SendMessage(g_hwCap,WM_CAP_SET_PREVIEWRATE,0,0);
               i1:=MessageBox(g_hwMain,'Ava katik ja pressi OK!','',MB_OK);
               g_Olukord:=5;
               g_Punkt:=1;
               g_T66Header.max1:=0;
               g_T66Header.nih1:=0;
               g_Sehkendan:=FALSE;
               SendMessage(g_hwCap,WM_CAP_SET_PREVIEWRATE,500,0);
               exit;
               end
          else begin  {p6hiline m66tmine}
               StrPCopy(g_T66Header.aeg,FormatDateTime('dd.mm.yy hh:mm',Now));
               StrLCopy(@buf1,g_DataPath,255);
               LStrCat(@buf1,g_T66Header.nimi);
               LStrCat(@buf1,g_T66Header.laiend);
               Salvesta(@buf1,@g_T66Header);
               Salvesta('fabry.tmp',@g_T66Header);
               g_Olukord:=6;
               PostMessage(g_hwMain,WM_COMMAND,202,0);
               exit;
               end;
          end;
     end;
803: begin  {faili kuvamine}
     g_Olukord:=2;  {fail avatud}
     with g_AvaHeader
     do begin
        if g_m1[max1]<>0 then ky1:=g_dy1/g_m1[max1];
        if g_m2[max2]<>0 then ky2:=g_dy2/g_m2[max2];
        if g_m3[max3]<>0 then ky3:=g_dy2/g_m3[max3];
        dx11:=g_dx1/punkte;
        dx21:=dx11;
        dx12:=50*dx11;
        dx22:=dx12;
        end;
     EnableMenuItem(g_hm1,101,MF_ENABLED);  {Open}
     EnableMenuItem(g_hm1,102,MF_ENABLED);  {Save As...}
     EnableMenuItem(g_hm1,103,MF_ENABLED);  {Close}
     EnableMenuItem(g_hm1,201,MF_ENABLED);  {M66da}
     EnableMenuItem(g_hm1,202,MF_GRAYED);  {Katkesta}
     StrPCopy(buf1,'Fabry95: ');
     StrCat(buf1,g_AvaHeader.nimi);
     SetWindowText(g_hwMain,buf1);
     SetWindowText(g_hwStatus,'Interferogramm');
```

```
                    InvalidateRect(g_hwMain,nil,TRUE);
                    PostMessage(g_hwMain,WM_PAINT,0,0);
                    exit;
                    end;
               804: begin
                    SetWindowText(g_hwStatus,'Initsialiseerimine...');
                    g_dx8:=g_VideoAken.x;  {videoakna koordinaadid}
                    if g_MaxX>g_dx8 then g_x8:=(g_MaxX-g_dx8) shr 1
                                    else begin
                                         g_dx8:=g_MaxX-50;
                                         g_x8:=25;
                                         end;
                    g_T66Header.punkte:=g_VideoAken.x;
                    g_T66Header.dx51:=g_dx8/g_VideoAken.x;
                    i1:=GetSystemMetrics(SM_CYMAXIMIZED);
                    r1:=i1*(g_Rida2-g_Rida1)/g_VideoAken.y;
                    g_dy8:={round(r1)}g_Rida2-g_Rida1;  {!!!!!}
                    g_y8:=(g_MaxY-g_dy8) shr 1;  {!!!!!}
                    g_y5:=g_y8-20;
                    g_y6:=g_MaxY-20;
                    g_dy5:=g_y5-20;
                    g_dy6:=g_y6-g_y8-g_dy8-20;
                    g_x1:=40;
                    g_x2:=40;
                    r1:=(g_MaxX-120)/3;
                    g_dx3:=round(r1);
                    g_dx4:=g_dx3;
                    g_dx1:=2*g_dx3;
                    g_dx2:=g_dx1;
                    g_x3:=80+g_dx1;
                    g_x4:=g_x3;
                    r1:=(g_MaxY-130)/2;
                    g_dy1:=round(r1);
                    g_dy2:=g_dy1;
                    g_dy3:=g_dy1;
                    g_dy4:=g_dy2;
                    g_y1:=40+g_dy1;
                    g_y2:=2*g_y1+20;
                    g_y3:=g_y1;
                    g_y4:=g_y2;
                    g_dy11:=g_dy1/10;
                    g_dy21:=g_dy2/10;
                    with g_DefaultHeader  {graafikuparameetrite arvutamine}
                    do begin
                       dx11:=g_dx1/g_VideoAken.x;
                       dx21:=dx11;
                       dx12:=50*dx11;
                       dx22:=dx12;
                       end;  {with}
                    Move(g_T66Header,g_AvaHeader,SizeOf(andmed));
                    SetWindowText(g_hwStatus,'Mõõtmiseks vali menüüst "Mõõda"');
                    exit;
                    end  {804}
               else begin MainProc:=DefWindowProc(g_hwMain,Msg,WParam,LParam); exit; end;
             end;  {case wParam}
             end;  {WM_COMMAND; peab tagastama 0, kui t66tleb}
WM_CREATE: begin
           g_hwMain:=hw1;
           g_Olukord:=1;  {tyhi aken}
           GetCurrentDirectory(255,@buf1);
           Move(buf1,g_MyraFail,255);
           p1:=LStrCat(@buf1,'\fabry95.ini');
           i1:=GetPrivateProfileString('GENERAL','DATAPATH','',@g_DataPath,255,p1);
           g_DataPath[i1]:='\';
           g_DataPath[i1+1]:=#0;
           i1:=GetPrivateProfileSection('6PETUS',@g_6petus,2000,p1);
           for i2:=1 to i1-1 do if g_6petus[i2]=0 then g_6petus[i2]:=13;
           i1:=GetPrivateProfileString('DEFAULT','TUGIFAIL','rublj.dat',@buf2[1],20,p1);
           buf2[0]:='\';
           LStrCat(g_MyraFail,buf2);
           g_Rida1:=GetPrivateProfileInt('GENERAL','RIDA1',100,p1);
           g_Rida2:=GetPrivateProfileInt('GENERAL','RIDA2',150,p1);
           g_VideoAken.x:=GetPrivateProfileInt('GENERAL','MAXX',768,p1);
           g_VideoAken.y:=GetPrivateProfileInt('GENERAL','MAXY',576,p1);
           with g_DefaultHeader
           do begin
              g_DefaultHeader.ridu:=g_Rida2-g_Rida1;
```

```
                g_DefaultHeader.punkte:=g_VideoAken.x;  {!!!!!}
                i1:=GetPrivateProfileString('GENERAL','FAILINIMI','',@nimi,19,p1);
                i1:=GetPrivateProfileString('GENERAL','LAIEND','.dat',@laiend[1],10,p1);
                laiend[0]:='.';
                pilte:=GetPrivateProfileInt('DEFAULT','KESKMISTADA',10,p1);
                i1:=GetPrivateProfileString('DEFAULT','KOMMENTAAR','',@komm,255,p1);
                k0:=GetPrivateProfileInt('GENERAL','K0',0,p1);
                k1:=GetPrivateProfileInt('GENERAL','K1',0,p1);
                k2:=GetPrivateProfileInt('GENERAL','K2',0,p1);
                k3:=GetPrivateProfileInt('GENERAL','K3',0,p1);
                k4:=GetPrivateProfileInt('GENERAL','K4',0,p1);
                k5:=GetPrivateProfileInt('GENERAL','K5',0,p1);
                k6:=GetPrivateProfileInt('GENERAL','K6',0,p1);
                k7:=GetPrivateProfileInt('GENERAL','K7',0,p1);
                k8:=GetPrivateProfileInt('GENERAL','K8',0,p1);
                end;
             g_hwStatus:=CreateStatusWindow(WS_CHILD+WS_VISIBLE,'',g_hwMain,hInstance);
             GetClientRect(g_hwStatus,TR1);
             g_dy7:=TR1.bottom;  {staatusakna k6rgus  ?????}
             g_MaxX:=GetSystemMetrics(SM_CXFULLSCREEN);
             g_MaxY:=GetSystemMetrics(SM_CYFULLSCREEN)-GetSystemMetrics(SM_CYMENU)-g_dy7;
             g_hm1:=GetMenu(g_hwMain);
             EnableMenuItem(g_hm1,102,MF_GRAYED);  {Save As...}
             EnableMenuItem(g_hm1,103,MF_GRAYED);  {Close}
             EnableMenuItem(g_hm1,104,MF_GRAYED);  {Print}
             EnableMenuItem(g_hm1,202,MF_GRAYED);  {Katkesta}
             g_DCMain:=GetDC(g_hwMain);
             g_hf0:=GetStockObject(ANSI_VAR_FONT);
             g_hf1:=GetStockObject(ANSI_VAR_FONT);
             g_hp0:=GetStockObject(BLACK_PEN);
             g_hp1:=CreatePen(PS_SOLID,2,0);
             g_hp2:=CreatePen(PS_SOLID,0,$f000);
             g_hp3:=CreatePen(PS_SOLID,0,$0f00);
             g_hp4:=CreatePen(PS_SOLID,0,$00f0);
             g_hb1:=GetStockObject(2);  {tyhi aken}
             g_hb2:=GetStockObject(COLOR_WINDOWFRAME);  {avatud fail}
             g_hb3:=GetStockObject(COLOR_BACKGROUND);  {h@@lestamine}
             SetBkMode(g_DCMain,TRANSPARENT);
             {}
             Move(g_DefaultHeader,g_AvaHeader,SizeOf(andmed));
             g_hBuf1:=GlobalAlloc(GMEM_MOVEABLE+GMEM_SHARE,2*768*576+10);
             g_pBuf1:=GlobalLock(g_hBuf1);
             PostMessage(g_hwMain,WM_COMMAND,804,0);
             SetTimer(g_hwMain,1,1000,nil);
             exit;
             end;  {peab tagastama 0}
   WM_CLOSE: begin
             i1:=SendMessage(g_hwCap,WM_CAP_SET_CALLBACK_FRAME,0,0);
             SendMessage(g_hwCap,WM_CAP_DRIVER_DISCONNECT,0,0);
             ReleaseDC(g_hwMain,g_DCMain);
             DeleteObject(g_hf1);
             DeleteObject(g_hp1);
             DeleteObject(g_hp2);
             DeleteObject(g_hp3);
             DeleteObject(g_hp4);
             Globalfree(g_hBuf1);
             KillTimer(g_hwMain,1);
             PostQuitMessage(0);
             exit;
             end  {peab tagastama 0}
   else begin MainProc:=DefWindowProc(hw1,Msg,WParam,LParam); exit; end;
end;  {case Message}
end;  {MainProc}

procedure WinMain;
begin
InitCommonControls;
with WindowClass do begin
                    cbClsExtra:=0;
                    cbWndExtra:=0;
                    hIcon:=0;
                    hCursor:=LoadCursor(0,IDC_ARROW);
                    style:=CS_CLASSDC+CS_BYTEALIGNCLIENT;
                    hInstance := 0;
                    lpfnWndProc:=@MainProc;
                    hbrBackground:=0;
                    lpszMenuName:='FABRY95';
```

```
                    lpszClassName:='FABRY95';
                    end;
if RegisterClass(WindowClass)=0 then begin piip; Halt(1); end;
with WClass2 do begin
                    cbClsExtra:=0;
                    cbWndExtra:=0;
                    hIcon:=0;
                    hCursor:=0;
                    style:=CS_PARENTDC;
                    hInstance := 0;
                    lpfnWndProc:=@TeineProc;
                    hbrBackground:=13;
                    lpszMenuName:=nil;
                    lpszClassName:='teine';
                    end;
RegisterClass(WClass2);
g_hwMain:=CreateWindow('FABRY95','Fabry-Perot'' etaloni uurimine',WS_OVERLAPPEDWINDOW,

cw_UseDefault,cw_UseDefault,cw_UseDefault,cw_UseDefault,0,0,HInstance,nil);
if g_hwMain=0 then begin
                  piip;
                  str(GetLastError,g_s1);
                  g_s1:=g_s1+#0;
                  g_i1:=MessageBox(0,@g_s1[1],'',MB_OK);
                  Halt(1);
                  end;
ShowWindow(g_hwMain,SW_SHOWMAXIMIZED);

while GetMessage(g_Msg1, 0, 0, 0) do begin
                                     TranslateMessage(g_Msg1);
                                     DispatchMessage(g_Msg1);
                                     end;
Halt(g_Msg1.wParam);
end;

begin
WinMain;
end.
```

Windows-programmi ini-fail

```
[GENERAL]
FAILINIMI=fabry
LAIEND=dat
DATAPATH=C:\PROGRA~1\FABRY95\DATA
MAXX=768
MAXY=576
RIDA1=270
RIDA2=370
K0=0
K1=0
K2=0
K3=0
K4=0
K5=0
K6=0
K7=0
K8=0

[DEFAULT]
KESKMISTADA=100
KOMMENTAAR=blaa-blaa
TUGIFAIL=rubla.dat

[6PETUS]
Justeerimiseks tuleb:
1. Asetada valgusallikas nii lдhedale sisendpilule kui vximalik, lьlitada sisse lambi toide
ja seada tццvooluks u. 50 mA. Avada monokromaatori katik ja seada tema trumli asendiks 2430°,
mis vastab elavhxbeda rohelise joone lainepikkusele (546,07 nm). Videoaknasse peab ilmuma
interferentsijoonte sьsteem, kusjuures sisendpilu kujutis ekraanil on horisontaalne.
Vдhendada sisendpilu laiust, kuni kujutis saab kitsamaks videoakna k6rgusest. Nihutada
interferentspilt monokromaatori trumli abil akna keskele; pilu servade kujutised peavad olema
paralleelsed akna pikema kьljega. Suurendada pilu laiust, kuni kujutis katab kogu videoakna
kxrguse.
2. Eemaldada ettevaatlikult kate Fabry-Perot' etaloni hoidjalt. Vдltida liigse valguse рддsemist
monokromaatorisse.
3. Reguleerida etalonihoidja liblikmutreid nii, et ekraanil oleksid nдha 5-10 interferentsijдrku
ning komponendid asetseksid risti akna pikema kьljega.
4. Kaamera nihutamisega piki siini teravustada kujutis.
5. Leida etaloni diafragma asend, mille puhul nxrgemad interferentsimaksimumid on kxige paremini
nдhtavad.
6. Korrata punkte 3-5 soovitud tulemuse saavutamiseni. Seejдrel asetada ettevaatlikult tagasi
etalonihoidja kate.
Nььd on seade justeeritud ja mxxtmisteks valmis.
```

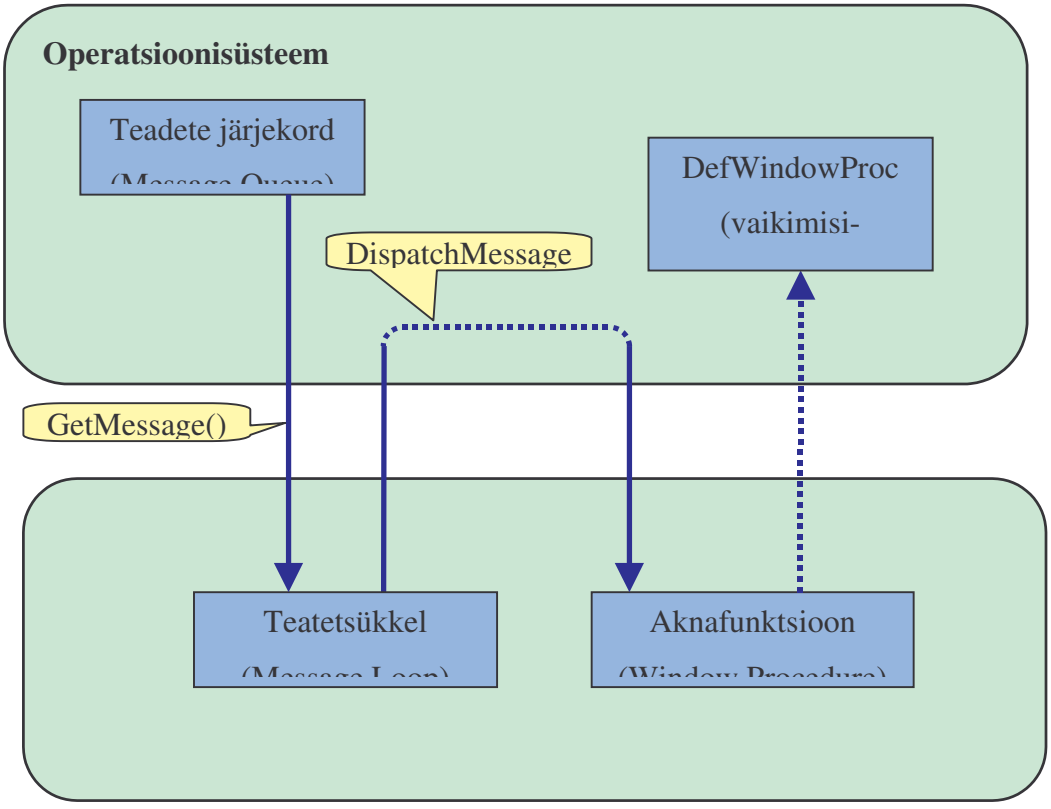
Andmefaili struktuur

```
Fabry95 andmefail
05.11.03 10:01
Keskmistatud kaadreid: 10
Ridu igas kaadris: 100
Piksleid igas reas: 768
Kommentaar: teine mootmine ilma vetilaatorita




2139368 655340
2435145 869180
1688132 355476
1446079 181518
1994439 312459
2251962 426071
   ...   ...
```

Lisa IV

Windows-programmi struktuur

Windowsi GUI-programmi (GUI - Graphical User Interface) põhimõisteteks on aken (Window), aknafunktsioon (Window Procedure) ja teatetsükkel (Message Loop). Kõige lihtsam definitsioon aknale on: ristkülikukujuline piirkond ekraanil, millele programm väljastab informatsiooni ja kuhu suunatakse sisestus kasutajalt. Kasutajaprogrammid (Applications) operatsioonisüsteemi MS Windows keskkonnas on kirjutatud lähtuvalt sündmuste-poolt-aetud (event driven) põhimõttest. Programm sisaldab reeglina ühte või mitut akent. Igale aknale vastab täisarvuline identifikaator ja kindla struktuuriga *callback*-funktsioon e. aknafunktsioon. Aknafunktsioon kutsutakse välja operatsioonisüsteemi poolt, kui leiab aset sündmus (event). Sündmusteks on näiteks klahvivajutused, hiire liikumine, katkestused kettakontrollerite ja portide poolt jne. Aknafunktsiooni sisendparameetriks on, lisaks akna identifikaatorile, spetsiaalne kirje *Message*, mis sisaldab informatsiooni sündmuse tüübi kohta. Akna loomine, aknafunktsiooni registreerimine ja teatetsükli realiseerimine toimub API-funktsioonide (API – Application Program Interface) abil. Programmi elutsükkel pärast akna loomist ja aknafunktsiooni registreerimist on kujutatud joonisel 1. Iga akna jaoks loob operatsioonisüsteem teadete



Joonis 1. Windows-programmi elutsükkel

järjekorra (Message Queue), kuhu paigutatakse saabumise järjekorras sündmustele vastavad *Message*-kirjed. Programm eraldab sealt järjekordse teate API-funktsiooni *GetMessage()* abil ja saadab ta *DispatchMessage()* abil (läbi operatsioonisüsteemi) oma peaakna aknafunktsioonile. Aknafunktsiooni piires sooritab programm vajalikud toimingud (peamiselt API-funktsioonide kutsungitena) ja tagastab juhtimise operatsioonisüsteemile. Kui programm teadet ei töötle, kutsutakse välja funktsioon *DefWindowProc()*, mis hoolitseb programmiakna vaikimisi-

```
program Programm1;

uses Windows, Messages;

{$R *.RES}

var wClass: TWndClass;
    Msg: TMsg;
    hwMain: HWND;

{aknafunktsioon:}
function AknaFunktsioon(hWnd,Msg,wParam,lParam: Longint): LongInt; stdcall;
begin
case Msg
of WM_CREATE: begin
              SetWindowText(hWnd,'Tere!!!');
              end;
   WM_DESTROY: begin
               PostQuitMessage(0);
               end
   else begin
        Result:=DefWindowProc(hWnd,Msg,wParam,lParam);
        end;
end;

{põhiprogrammi algus}
begin
wClass.lpszClassName:='CN';
wClass.lpfnWndProc:=@AknaFunktsioon;
wClass.hInstance:=hInstance;
wClass.hbrBackground:=1;
RegisterClass(wClass);

{programmi põhiakna loomine:}
hwMain:=CreateWindow(wClass.lpszClassName,'Title Bar',
                     WS_OVERLAPPEDWINDOW or WS_VISIBLE,
                     10,10,340,220,0,0,hInstance,nil);

{teadete töötlemise tsükkel:}
while GetMessage(Msg,0,0,0)
do begin
   DispatchMessage(Msg);
   end;

end.
```

aknafunktsioon

teatetsükkel

Joonis 2. Lihtsa Windows-programmi näide

toimingute eest. *DefWindowProc()* kutsutakse välja samade parameetritega, mis aknafunktsioongi. API-de kasutamisel ei sõltu Windows-programmi struktuur programmeerimiskeelest. Vaatleme Windows-programmi lihtsaimat näidet keele Pascal/Delphi baasil (joonis 2). Programmi alguses registreeritakse akna klass API-funktsiooniga *RegisterClass()*. Selle funktsiooni parameetriks on *TWndClass*-tüüpi kirjemuutuja, mille väljad määravad akna peamised omadused, nagu paiknemine ekraanil, tausta värv, hiirekursori kuju jne. Väli *lpfnWndProc* määrab aknafunktsiooni. API-funktsiooniga *CreateWindow()* luuakse programmi peamine aken, see funktsioon tagastab loodud akna identifikaatori. Järgneb teadete töötlemise tsükkel: API-funktsioon *GetMessage()* võtab järjekorrast (Message Queue) järgmise teate ja *DispatchMessage()* saadab selle teate aknafunktsioonile. Kirje *Message* sisaldab tegelikult rohkem väljasid, aga aknafunktsioonile antakse parameetriteks ainult akna identifikaator, teate tüüp ning kaks täisarvulist parameetrit: *wParam* ja *lParam*. Nende parameetrite tähendus sõltub teate tüübist. Joonisel 2 toodud näites töötleb *AknaFunktsioon()* ainult kahte teadet: *WM_CREATE* ja *WM_DESTROY*. Ülejäänud teadete puhul kutsutakse välja API-funktsioon *DefWindowProc()* ja antakse juhtimine tagasi operatsioonisüsteemile.

WM_CREATE on esimene teade, mille operatsioonisüsteem programmi aknafunktsioonile saadab. Näites seatakse selle teate töötlemise käigus peaakna tiiteltekstiks "Tere!!!", kasutades API-funktsiooni *SetWindowText()*. Teade *WM_DESTROY* saadetakse aknafunktsioonile pärast akna sulgemist. Selle teate töötlemise käigus kutsutakse välja *WM_QUIT* postitamine API-funktsiooni *PostQuitMessage()* abil. Funktsioon *GetMessage()* tagastab *WM_QUIT* saabumisel väärtuse *FALSE* (kõikide teiste teadete puhul *TRUE*), *while*-tsükkel katkeb ja koos sellega ka programmi elutsükkel.

API-funktsioonide ja Windows GUI-programmeerimise kohta võib lugeda [10].