Lampiran I

Program

```
//*********************************//
// InOut.hpp                       //
// Copyright 1999, Rahmat Sobari   //
// Compiled with Borland C++ 5.0   //
//*********************************//

#ifndef INOUT_H
#define INOUT_H
#define cxNewWidth  441
#define cyNewHeight 461
class TKolektor{
  public :
    TKolektor(unsigned ,unsigned ,unsigned );
    void CreateGrid(TDC&);
    void OutData(unsigned , unsigned , bool);
    unsigned char InData(unsigned );
    unsigned GetVCEData();
    void SendVBBData(unsigned );
    void SendVCCData(unsigned );
    void PlotXY(TDC&);

  protected :
    unsigned DataPort;
    unsigned StatusPort;
    unsigned ControlPort;
    int x0;
    int y0;
    bool ErrINTR;
};
#endif
```

```
//*********************************//
// InOut.cpp                       //
// Copyright 1999, Rahmat Sobari   //
// Compiled with Borland C++ 5.0   //
//*********************************//

#include <owl/dc.h>
#pragma hdrstop
#include "InOut.hpp"
#pragma inline

TKolektor::TKolektor(unsigned Data, unsigned Status, unsigned Control){
  DataPort = Data;
  StatusPort = Status;
  ControlPort = Control;
  x0 = 110;
  y0 = 350;
  ErrINTR = false;
}

void TKolektor::CreateGrid(TDC& dc){
  int i;
  TPoint Pt1(0,0);
  TPoint Pt2(0,0);
  char* VCE[] = {"-2","-1","0 ","1 ","2 ","3 ","4 ","5 "};
  char* IC[] = {"600 ","500 ","400 ","300 ","200 ","100 ","0","-100","-200"};

  TPen GrayPen((HPEN)CreatePen(PS_SOLID, 0, RGB(192,192,192)), AutoDelete);
  dc.SetROP2(R2_COPYPEN);
  dc.SetBkMode(TRANSPARENT);
  dc.SelectObject(GrayPen);
  Pt1.x = 0; Pt2.x = cxNewWidth;    //Garis horizontal
  for(Pt1.y = 0; Pt1.y <= cyNewHeight; Pt1.y+=10){
    Pt2.y = Pt1.y;
    dc.MoveTo(Pt1);
    dc.LineTo(Pt2);
  }
  Pt1.y = 0; Pt2.y = cyNewHeight;   //Garis vertikal
  for(Pt1.x = 0; Pt1.x <= cxNewWidth; Pt1.x+=10){
    Pt2.x = Pt1.x;
    dc.MoveTo(Pt1);
    dc.LineTo(Pt2);
  }
  TPen BluePen((HPEN)CreatePen(PS_SOLID, 1, RGB(0,0,255)), AutoDelete);
  TBrush NullBrush((HBRUSH)GetStockObject(NULL_BRUSH), AutoDelete);
  dc.SelectObject(BluePen);
  dc.SelectObject(NullBrush);
  dc.Rectangle(TPoint(0,0), TPoint(x0,y0));
  dc.Rectangle(TPoint(x0,0), TPoint(cxNewWidth,y0));
  dc.Rectangle(TPoint(0,y0), TPoint(x0,cyNewHeight));
  dc.Rectangle(TPoint(x0,y0), TPoint(cxNewWidth,cyNewHeight));
  Pt1.y = y0+1;
  for(i = 0; i < 8; i++){
    if(i < 2) Pt1.x = i*50+2;
    if(i == 2) Pt1.x = i*50+12;
    if(i >= 3) Pt1.x = i*50+7;
    dc.TextOut(Pt1.x, Pt1.y, VCE[i], 2);
  }
  dc.TextOut(Pt1.x+40, Pt1.y, "VCE(V)", 6);
  Pt1.x = 80+4;
  for(i = 0; i < 9; i++){
    if(i == 6){
      Pt1.x = 80;
      i++;
    }
    Pt1.y = i*50+43;
```

```
      dc.TextOut(Pt1.x, Pt1.y, IC[i], 4);
    }
    dc.TextOut(Pt1.x-11, 0, "IC(uA)", 6);
  }

  void TKolektor::PlotXY(TDC& dc){
    unsigned VBBData[] = {0x15, 0x3a, 0x55, 0x70, 0x8b};
    float VCEConst = 4.92/255.0;
    float VCCConst = 4.98/255.0;
    float ICConst  = 1000.0/7.35;
    unsigned VCEData;
    float VCC, VCE, IC;
    int x, y;
    char Buffer[10];

    for(int i = 0; i < 5; i++){
      OutData(ControlPort,0x08,true); //00001000b CLK off (bit 3)
      OutData(ControlPort,0x00,false);//00000000b CLK on (bit 3)
      SendVBBData(VBBData[i]);
      for(unsigned VCCData = 0x00; VCCData <= 0xff; VCCData++){
        SendVCCData(VCCData);
        VCEData = (GetVCEData()+GetVCEData()+GetVCEData())/3;
        if(ErrINTR){
          ErrINTR = false;
          return;
        }
        VCE = VCEConst*VCEData;
        x = x0+VCEData;
        if(VCE){
          y = y0-((160/VCE)/2);
          dc.MoveTo(TPoint(x,y));      // Kurva disipasi daya
          dc.LineTo(TPoint(x,y));
        }
        VCC = VCCConst*VCCData;
        IC = ICConst*(VCC-VCE);
        y = y0-(IC/2.0);
        dc.MoveTo(TPoint(x,y));        // Kurva kolektor
        dc.LineTo(TPoint(x,y));
        if((VCCData == 0xff) && (VBBData[i] == 0x15))
          dc.TextOut(140, 2, "P = 160 uW", 10);
      }
      wsprintf(Buffer, "IB = %d uA", i);
      if(VBBData[i] != 0x15) dc.TextOut(x+10, y-7, Buffer, 9);
    }
  }

  unsigned TKolektor::GetVCEData(){
    int i = 0;
    unsigned char Hasil;
    unsigned char Temp;
    unsigned char INTR;

    OutData(ControlPort,0x09,false);  //00001001b  Reset ADC (set bit 0 dan 3)
    OutData(ControlPort,0x00,false);  //00000000b  Konversikan ke digital
    do{
      i++;
      INTR = InData(StatusPort);
      INTR &= 0x80;                   //10000000b  Set bit 7, reset bit lainnya
      if((i == 0x0ffff) && (INTR != 0x80)){
        MessageBox(0, "INTR tidak bisa set", "Kolektor", MB_ICONHAND);
        ErrINTR = true;
        return 0;
      }
    }while(INTR != 0x80);             //Selama INTR (bit 7) tidak set, tunggu
    OutData(ControlPort,0x01,false);  //00000001b  Buka jalur 4 bit MSB (bit 2)
    Hasil = InData(StatusPort);       //Simpan 4 bit MSB hasil konversi
```

```
  Hasil &= 0x78;                          //0XXXX000b Reset selain bit 3-6
  Hasil <<= 1;                            //Geser 1 bit ke kiri
  OutData(ControlPort,0x05,false);        //00000101b  Buka jalur 4 bit LSB (bit 2)
  Temp = InData(StatusPort);              //Simpan 4 bit LSB hasil konversi
  Temp &= 0x78;
  Temp >>= 3;                             //Geser 3 bit ke kanan
  Hasil += Temp;                          //Gabungkan menjadi 8 bit
  return (unsigned)Hasil;
}

void TKolektor::SendVBBData(unsigned Data){
  OutData(DataPort,Data,false);           //Kirim data
  OutData(ControlPort,0x02,true);         //00000010b Buka jalur DAC-B (bit 1)
  OutData(ControlPort,0x00,false);        //00000000b Tutup jalur DAC-B (bit 1)
}
void TKolektor::SendVCCData(unsigned Data){
  OutData(DataPort,Data,false);           //Kirim data
  OutData(ControlPort,0x01,true);         //00000001b Buka jalur DAC-C (bit 0)
  OutData(ControlPort,0x00,false);        //00000000b Tutup jalur DAC-C (bit 0)
}

void TKolektor::OutData(unsigned PortNumber, unsigned Data, bool Wait){
  asm{
    push dx                               //Simpan nilai register dx ke stack
    push ax                               //Simpan nilai register ax ke stack
    mov dx,word ptr [PortNumber]          //Isi register dx dengan nomor port
    mov al,byte ptr [Data]                //Isi register ax dengan data
    out dx,al                             //Kirim data
    pop ax                                //Kembalikan nilai register dx
    pop dx                                //Kembalikan nilai register ax
  }
  if(Wait) for(int i = 0; i < 0x0ffff; i++){}
}

unsigned char TKolektor::InData(unsigned PortNumber){
  unsigned char Hasil = 0;
  asm{
    push dx
    push ax
    mov dx,word ptr [PortNumber]
    in al,dx                              //Simpan data dari port di register al
    mov [Hasil],al
    pop ax
    pop dx
  }
  return Hasil;
}
```

```
//*********************************//
// Kolektor.cpp                    //
// Copyright 1999, Rahmat Sobari   //
// Compiled with Borland C++ 5.0   //
//*********************************//

#include <owl/applicat.h>
#include <owl/decframe.h>
#include <owl/dc.h>
#include <owl/opensave.h>
#include <owl/controlb.h>
#include <owl/buttonga.h>
#include <classlib/file.h>
#pragma hdrstop
#include "InOut.hpp"

const int COMMANDS      = 100;
const int CM_FILENEW    = 200;
const int CM_FILEOPEN   = 201;
const int CM_FILESAVE   = 202;
const int CM_FILESAVEAS = 203;
const int CM_ABOUT      = 204;
const int CM_CURVE      = 205;
const int CM_FILECLOSE  = 206;
const int IDD_ABOUT     = 300;

class TKolektorWnd : public TWindow{
  public :
    TKolektorWnd(TWindow* = 0);
   ~TKolektorWnd(){
      if(DIB) delete DIB;
      delete FileData;
      delete Kolektor;'
    }

  protected :
    TDC* DragDC;
    TDib* DIB;
    TPoint* BeginPt;
    TPoint* EndPt;
    TKolektor* Kolektor;
    TOpenSaveDialog::TData* FileData;
    bool IsDirty, IsNewFile, IsClose;

    void SetupWindow();
    void CreateBitmap(TDC&, TRect&);
    bool CanClose();
    void EvLButtonDown(uint, TPoint&);
    void EvMouseMove(uint, TPoint&);
    void EvLButtonUp(uint, TPoint&);
    void EvKeyDown(uint, uint, uint);
    void Paint(TDC&, bool, TRect&);
    void CmFileNew();
    void CmFileOpen();
    void CmFileSave();
    void CmFileSaveAs();
    void CmAbout(){TDialog(this, IDD_ABOUT).Execute();}
    void CmCurve();
    void CmFileClose();
    void SaveFile();
    void OpenFile();

  DECLARE_RESPONSE_TABLE(TKolektorWnd);
};

DEFINE_RESPONSE_TABLE1(TKolektorWnd, TWindow)
```

```
  EV_WM_LBUTTONDOWN,
  EV_WM_RBUTTONDOWN,
  EV_WM_MOUSEMOVE,
  EV_WM_LBUTTONUP,
  EV_WM_KEYDOWN,
  EV_COMMAND(CM_FILENEW, CmFileNew),
  EV_COMMAND(CM_FILEOPEN, CmFileOpen),
  EV_COMMAND(CM_FILESAVE, CmFileSave),
  EV_COMMAND(CM_FILESAVEAS, CmFileSaveAs),
  EV_COMMAND(CM_ABOUT, CmAbout),
  EV_COMMAND(CM_CURVE, CmCurve),
  EV_COMMAND(CM_FILECLOSE, CmFileClose),
END_RESPONSE_TABLE;

TKolektorWnd::TKolektorWnd(TWindow* parent):TWindow(parent, 0, 0){
  DragDC = 0;
  DIB = 0;
  BeginPt = 0;
  EndPt = 0;
  Attr.W = cxNewWidth;
  Attr.H = cyNewHeight;
  Attr.ExStyle = WS_EX_CLIENTEDGE;
  IsNewFile = true;
  IsDirty   = false;
  Kolektor = new TKolektor(0x378, 0x379, 0x37a);
  FileData  = new TOpenSaveDialog::TData(OFN_HIDEREADONLY|OFN_FILEMUSTEXIST,
                                         "Asv Files (*.ASV)|*.asv|", 0, "",
                                         "ASV");
}

void TKolektorWnd::SetupWindow(){
  TWindow::SetupWindow();
  SetBkgndColor(GetSysColor(COLOR_APPWORKSPACE));
  IsClose = true;
}

void TKolektorWnd::EvLButtonDown(uint, TPoint& Pt){
  if(!DragDC && !BeginPt && !EndPt && !IsClose){
    SetCapture();
    BeginPt = new TPoint(Pt);
    EndPt = new TPoint(Pt);
    DragDC = new TClientDC(HWindow);
    DragDC->SetROP2(R2_NOT);
    DragDC->SetBkMode(TRANSPARENT);
    DragDC->MoveTo(Pt);
    DragDC->LineTo(Pt);
    IsDirty = true;
  }
}

void TKolektorWnd::EvMouseMove(uint, TPoint& Pt){
  if(DragDC && BeginPt && EndPt){
    DragDC->MoveTo(*BeginPt);
    DragDC->LineTo(*EndPt);
    DragDC->MoveTo(*BeginPt);
    DragDC->LineTo(Pt);
    *EndPt = Pt;
  }
}

void TKolektorWnd::EvLButtonUp(uint, TPoint& Pt){
  if(DragDC && BeginPt && EndPt){
    DragDC->MoveTo(*BeginPt);
    DragDC->LineTo(*EndPt);
    DragDC->SetROP2(R2_COPYPEN);
    DragDC->MoveTo(*BeginPt);
```

```
    DragDC->LineTo(Pt);
    ReleaseCapture();
    TRect Rect(GetClientRect());
    CreateBitmap(*DragDC, Rect);
    delete DragDC;
    DragDC = 0;
    delete BeginPt;
    BeginPt = 0;
    delete EndPt;
    EndPt = 0;
  }
}

void TKolektorWnd::EvKeyDown(uint key, uint, uint){
  if(key == VK_ESCAPE) Invalidate(false);
}

void TKolektorWnd::Paint(TDC& dc, bool, TRect&){
  if(DIB){
    dc.SetDIBitsToDevice(GetClientRect(), TPoint(0,0), *DIB);
  }
}

void TKolektorWnd::CreateBitmap(TDC& dc, TRect& Rect){
  TBitmap Bitmap(dc, Rect.Width(), Rect.Height());
  TMemoryDC MemDC(dc);
  MemDC.SelectObject(Bitmap);
  MemDC.BitBlt(Rect, dc, TPoint(0,0));
  if(DIB) delete DIB;
  DIB = new TDib(Bitmap);
}

bool TKolektorWnd::CanClose(){
  if(IsDirty)
    switch (MessageBox("Apakah anda ingin menyimpan?", "Kolektor",
                       MB_YESNOCANCEL | MB_ICONQUESTION)){
      case IDCANCEL: return false;
      case IDYES: CmFileSave();
    }
  return true;
}

void TKolektorWnd::CmFileNew(){
  if(CanClose()){
    TClientDC dc(HWindow);
    TRect Rect(GetClientRect());
    dc.PatBlt(Rect, WHITENESS);
    Kolektor->CreateGrid(dc);
    CreateBitmap(dc, Rect);
    IsDirty = IsClose = false;
    IsNewFile = true;
  }
}

void TKolektorWnd::CmFileOpen(){
  if(CanClose())
    if((TFileOpenDialog(this, *FileData)).Execute() == IDOK) OpenFile();
}

void TKolektorWnd::CmFileSave(){
  if(!IsClose){
    if(IsNewFile) CmFileSaveAs();
    else SaveFile();
  }
}
```

```
void TKolektorWnd::CmFileSaveAs(){
  if(!IsClose){
    if(IsNewFile) strcpy(FileData->FileName, "");
    if((TFileSaveDialog(this, *FileData)).Execute() == IDOK) SaveFile();
  }
}

void TKolektorWnd::SaveFile(){
  if(DIB){
    DIB->WriteFile(FileData->FileName);
    IsDirty = IsNewFile = false;
  }
}

void TKolektorWnd::OpenFile(){
  TFile File(FileData->FileName, TFile::ReadWrite);
  if(DIB) delete DIB;
  DIB = new TDib(File);
  IsNewFile = IsDirty = IsClose = false;
  Invalidate(false);
}

void TKolektorWnd::CmFileClose(){
  if(CanClose()){
    if(DIB) delete DIB;
    DIB = 0;
    IsClose = true;
    IsDirty = false;
    SetBkgndColor(GetSysColor(COLOR_APPWORKSPACE));
    Invalidate(true);
  }
}

void TKolektorWnd::CmCurve(){
  if(!DragDC && !IsClose){
    TPen PenGraph((HPEN)CreatePen(PS_SOLID, 2, RGB(0,0,0)), AutoDelete);
    DragDC = new TClientDC(HWindow);
    DragDC->SelectObject(PenGraph);
    Kolektor->PlotXY(*DragDC);
    CreateBitmap(*DragDC, GetClientRect());
    delete DragDC;
    DragDC = 0;
    IsDirty = true;
  }
}

class TKolektorApp : public TApplication{
  public:
    TKolektorApp() : TApplication() {}
    void InitMainWindow();
};

void TKolektorApp::InitMainWindow(){
  TDecoratedFrame* frame = new TDecoratedFrame(0,"Kolektor",new TKolektorWnd,true);
  TControlBar* cb = new TControlBar(frame);
  cb->Insert(*new TButtonGadget(CM_FILENEW,CM_FILENEW,TButtonGadget::Command));
  cb->Insert(*new TButtonGadget(CM_FILEOPEN,CM_FILEOPEN,TButtonGadget::Command));
  cb->Insert(*new TButtonGadget(CM_FILESAVE,CM_FILESAVE,TButtonGadget::Command));
  cb->Insert(*new TButtonGadget(CM_FILESAVEAS,CM_FILESAVEAS,TButtonGadget::Command));
  cb->Insert(*new TSeparatorGadget);
  cb->Insert(*new TButtonGadget(CM_ABOUT,CM_ABOUT, TButtonGadget::Command));
  frame->Insert(*cb, TDecoratedFrame::Top);
  SetMainWindow(frame);
  TFrameWindow* MainWnd = GetMainWindow();
  MainWnd->AssignMenu(COMMANDS);
  MainWnd->Attr.W = cxNewWidth + 10;
```

```
  MainWnd->Attr.H = cyNewHeight + 80;
  MainWnd->Attr.Style &= ~WS_MAXIMIZEBOX;
  MainWnd->Attr.Style &= ~WS_THICKFRAME;
}

int OwlMain(int , char* []){
  return TKolektorApp().Run();
}
```

```
//*****************************************//
// Kolektor.rc                             //
// Produced by Borland Resource Workshop   //
// Edited by Rahmat Sobari                 //
//*****************************************//

#ifdef RC_INVOKED
  #ifndef WORKSHOP_INVOKED
    # include <windows.h>
  #endif

#define COMMANDS      100
#define CM_FILENEW    200
#define CM_FILEOPEN   201
#define CM_FILESAVE   202
#define CM_FILESAVEAS 203
#define CM_ABOUT      204
#define CM_CURVE      205
#define CM_FILECLOSE  206
#define IDD_ABOUT     300

COMMANDS MENU
{
 POPUP "&Berkas"
 {
  MENUITEM "B&aru",              CM_FILENEW
  MENUITEM "B&uka",              CM_FILEOPEN
  MENUITEM "&Simpan",            CM_FILESAVE
  MENUITEM "S&impan sebagai",    CM_FILESAVEAS
  MENUITEM "&Tutup",             CM_FILECLOSE
  MENUITEM SEPARATOR
  MENUITEM "&Gambar Kurva",      CM_CURVE
  MENUITEM SEPARATOR
  MENUITEM "&Keluar",            24310
 }
 POPUP "&Bantu"
 {
  MENUITEM "&Tentang",           CM_ABOUT
 }
}

IDD_ABOUT DIALOG 37, 25, 170, 95
STYLE DS_3DLOOK | DS_CENTER | WS_POPUP | WS_VISIBLE | WS_CAPTION | WS_SYSMENU
CAPTION "Kolektor"
FONT 8, "MS Sans Serif"
{
 DEFPUSHBUTTON "OK", IDOK, 60, 73, 50, 14
 CTEXT "Penampil Kurva Kolektor Transistor", -1, 12, 39, 150, 10
 CTEXT "\251 1999 Rahmat Sobari", -1, 10, 56, 150, 8
 ICON "IDI_TUTORIAL", -1, 78, 11, 14, 16
}

IDI_TUTORIAL ICON
{
 '00 00 01 00 01 00 20 20 10 00 00 00 00 00 E8 02'
 '00 00 16 00 00 00 28 00 00 00 20 00 00 00 40 00'
 '00 00 01 00 04 00 00 00 00 00 80 02 00 00 00 00'
 '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 80 00 00 80 00 00 00 80 80 00 80 00'
 '00 00 80 00 80 00 80 80 00 00 80 80 80 00 C0 C0'
 'C0 00 00 00 FF 00 00 FF 00 00 00 FF FF 00 FF 00'
 '00 00 FF 00 FF 00 FF FF 00 00 FF FF FF 00 00 00'
 '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 00 70 07 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 77 77 77 77 00 00 00 00 00 00 00 00'
```

```
  '00 00 00 00 00 00 77 77 00 00 00 00 00 00 00 00'
  '00 00 00 00 77 77 00 00 00 00 00 00 00 00 00 00'
  '00 00 00 00 77 77 77 77 00 00 00 00 00 00 00 00'
  '00 00 00 00 00 00 77 77 00 00 00 00 00 00 00 00'
  '00 00 00 00 77 77 00 00 00 00 00 00 00 00 00 00'
  '00 00 00 00 77 77 77 77 00 00 00 00 00 00 00 00'
  '00 00 00 00 0F 0F F0 F0 00 00 00 00 00 00 00 00'
  '00 00 00 00 0F 0F F0 F0 00 00 00 00 00 00 00 00'
  '00 00 00 00 0F 0F F0 F0 00 00 00 00 00 00 00 00'
  '00 00 00 00 FF 0F F0 FF 00 00 00 00 00 00 00 00'
  '00 00 00 0F FF 0F F0 FF F0 00 00 00 00 00 00 00'
  '00 00 00 FF FF 0F F0 FF FF 00 00 00 00 00 00 00'
  '00 00 0F FF FF 0F F0 FF FF F0 00 00 00 00 00 00'
  '00 00 0F FF FF 0F F0 FF FF F0 00 00 00 00 00 00'
  '00 00 FF FF FF 9B 9B FF FF FF 00 00 00 00 00 00'
  '00 00 FF FF FF B9 B9 FF FF FF 00 00 00 00 00 00'
  '00 00 FF FF FF FF FF FF FF FF 00 00 00 00 00 00'
  '00 00 FF FF FF FF FF FF FF FF 00 00 00 00 00 00'
  '00 00 FF FF FF FF FF FF FF FF 00 00 00 00 00 00'
  '00 00 0F FF FF FF FF FF FF F0 00 00 00 00 00 00'
  '00 00 0F FF FF FF FF FF FF F0 00 00 00 00 00 00'
  '00 00 00 FF FF FF FF FF FF 00 00 00 00 00 00 00'
  '00 00 00 0F FF FF FF FF F0 00 00 00 00 00 00 00'
  '00 00 00 00 0F FF FF F0 00 00 00 00 00 00 00 00'
  '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00'
  '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00'
  '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00'
  '00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF'
  'FF FF FF FE 7F FF FF FC 3F FF FF F0 0F FF FF F0'
  '0F FF FF F0 0F FF FF F0 0F FF FF F0 0F FF FF F0'
  '0F FF FF F0 0F FF FF F0 0F FF FF F0 0F FF FF F0'
  '0F FF FF E0 07 FF FF C0 03 FF FF 80 01 FF FF 00'
  '00 FF FF 00 00 FF FE 00 00 7F FE 00 00 7F FE 00'
  '00 7F FE 00 00 7F FE 00 00 7F FF 00 00 FF FF 00'
  '00 FF FF 80 01 FF FF C0 03 FF FF E0 07 FF FF F8'
  '1F FF FF FF FF FF FF FF FF FF FF FF FF FF FF'
}

CM_FILENEW BITMAP LOADONCALL MOVEABLE
{
  '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
  '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
  '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
  '00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 80'
  '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
  '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
  '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
  '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
  '40 00 88 88 88 88 88 88 88 88 88 88 90 00 88 88'
  '88 88 88 88 88 88 88 88 20 00 88 88 88 88 88 88'
  '88 88 88 88 40 18 88 88 88 88 88 88 88 88 88 88'
  '00 00 88 88 88 88 88 88 88 88 88 88 00 00 88 88'
  '88 88 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
  '88 88 88 88 0B 00 88 80 08 88 88 88 88 88 88 88'
  '00 00 88 80 08 88 88 88 88 88 88 88 00 00 88 80'
  '08 88 88 84 44 44 44 48 00 00 88 80 08 80 88 84'
  'EF EF EF 48 0B 00 88 80 07 80 08 84 FE FE FE 48'
  '00 00 88 87 00 00 00 84 EF EF EF 48 00 00 88 88'
  '70 00 00 84 FE FE FE 48 00 00 88 88 88 80 08 84'
  'EF EF EF 48 0B 00 88 88 88 80 88 84 FE FE 44 48'
  'FF FF 88 88 88 88 88 84 EF EF 44 88 FF FF 88 88'
  '88 88 88 84 44 44 48 88 00 00 88 88 88 88 88 88'
  '88 88 88 88 0B 00'
}

CM_FILEOPEN BITMAP LOADONCALL MOVEABLE
```

```
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '46 00 80 00 00 00 00 00 88 88 88 88 00 00 80 87'
 '77 77 77 70 88 88 88 88 11 02 80 F8 88 88 88 70'
 '88 88 88 88 08 00 80 F9 98 88 88 70 88 88 88 88'
 '46 00 80 FF FF FF FF 80 88 88 88 88 00 00 80 00'
 '00 00 00 00 88 88 88 88 11 00 88 88 88 88 88 88'
 '88 88 88 88 08 30 88 80 08 88 88 88 88 88 88 88'
 '46 00 88 80 08 88 88 88 88 88 88 88 00 00 88 80'
 '08 88 88 84 44 44 44 48 11 00 88 80 08 80 88 84'
 'EF EF EF 48 08 00 88 80 07 80 08 84 F4 44 4E 48'
 '00 00 88 87 00 00 00 84 EF EF EF 48 55 55 88 88'
 '70 00 00 84 F4 44 4E 48 00 00 88 88 88 80 08 84'
 'EF EF EF 48 88 88 88 88 88 80 88 84 F4 4E 44 48'
 '00 00 88 88 88 88 88 84 EF EF 44 88 00 00 88 88'
 '88 88 88 84 44 44 48 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 00 00'
}

CM_FILESAVE BITMAP LOADONCALL MOVEABLE
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '46 00 88 88 88 88 00 00 00 00 00 08 00 00 88 88'
 '88 88 08 77 77 77 77 08 11 00 88 88 88 88 0F 88'
 '88 88 87 08 08 30 88 88 88 88 0F 99 88 88 87 08'
 '46 00 88 88 88 88 0F FF FF FF F8 08 00 00 88 88'
 '88 88 00 00 00 00 00 08 11 00 88 88 88 88 88 88'
 '88 88 88 88 08 00 88 88 88 88 88 88 88 00 88 88'
 '00 00 88 88 88 88 88 88 80 00 08 88 55 55 84 44'
 '44 44 48 88 00 00 00 88 00 00 84 EF EF EF 48 88'
 '88 00 88 88 88 88 84 F4 44 4E 48 88 88 00 88 88'
 '00 00 84 EF EF EF 48 88 87 00 88 88 00 00 84 F4'
 '44 4E 48 00 00 07 88 88 00 00 84 EF EF EF 48 00'
 '00 78 88 88 00 00 84 F4 4E 44 48 88 88 88 88 88'
 '00 00 84 EF EF 44 88 88 88 88 88 88 00 00 84 44'
 '44 48 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 11 11'
}

CM_FILESAVEAS BITMAP LOADONCALL MOVEABLE
{
 '42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
 '00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
 '00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
 '00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 80'
 '00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
 '00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
 '00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
 '00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
 '46 00 88 88 88 88 88 84 44 44 44 48 00 00 88 88'
 '88 88 88 84 FF FF FF 48 11 00 88 88 88 88 88 84'
 'F4 44 4F 48 08 00 88 88 88 88 88 84 FF FF FF 48'
```

```
 '00 00 88 88 88 88 88 84 F4 44 4F 48 55 55 88 88'
 '88 88 88 84 FF FF FF 48 00 00 88 88 88 88 88 84'
 'F4 4F 44 48 88 88 88 88 88 88 88 84 FF FF 44 88'
 '00 00 88 88 88 88 88 84 44 44 48 88 00 00 84 44'
 '44 44 48 88 88 88 88 88 00 00 84 EF EF EF 48 88'
 '88 00 88 88 00 00 84 F4 44 4E 48 88 80 00 08 88'
 '00 00 84 EF EF EF 48 88 00 00 00 88 00 00 84 F4'
 '44 4E 48 88 88 00 88 88 00 00 84 EF EF EF 48 88'
 '87 00 88 88 11 11 84 F4 4E 44 48 00 00 07 88 88'
 '00 00 84 EF EF 44 88 00 00 78 88 88 00 00 84 44'
 '44 48 88 88 88 88 88 88 00 00 88 88 88 88 88 88'
 '88 88 88 88 0B 00'
}

CM_ABOUT BITMAP
{
'42 4D 66 01 00 00 00 00 00 00 76 00 00 00 28 00'
'00 00 14 00 00 00 14 00 00 00 01 00 04 00 00 00'
'00 00 F0 00 00 00 00 00 00 00 00 00 00 00 00 00'
'00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 80'
'00 00 00 80 80 00 80 00 00 00 80 00 80 00 80 80'
'00 00 80 80 80 00 C0 C0 C0 00 00 00 FF 00 00 FF'
'00 00 00 FF FF 00 FF 00 00 00 FF 00 FF 00 FF FF'
'00 00 FF FF FF 00 88 88 88 88 88 88 88 88 88 88'
'00 00 88 88 88 84 44 44 48 88 88 88 00 00 88 88'
'84 44 44 44 44 48 88 88 00 00 88 88 44 46 FF 64'
'44 44 88 88 00 00 88 84 44 4F FF F6 44 44 48 88'
'00 00 88 44 44 4F F6 48 44 44 44 88 00 00 88 44'
'44 46 FF 44 44 44 44 88 01 01 84 44 44 44 FF 64'
'44 44 44 48 08 33 84 44 44 44 6F F4 44 44 44 48'
'00 00 84 44 44 44 4F F6 44 44 44 48 00 00 84 44'
'44 48 46 FF 44 44 44 48 00 00 84 44 44 46 FF FF'
'44 44 44 48 00 00 84 44 44 44 6F F6 44 44 44 48'
'00 00 88 44 44 44 44 44 44 44 44 88 00 00 88 44'
'44 44 44 66 44 44 44 88 00 00 88 84 44 44 46 FF'
'64 44 48 88 00 00 88 88 44 44 46 FF 64 44 88 88'
'00 00 88 88 84 44 44 66 44 48 88 88 00 00 88 88'
'88 84 44 44 48 88 88 88 00 00 88 88 88 88 88 88'
'88 88 88 88 00 00'
}
#endif
```

```
#include <owl/applicat.h>
#include <owl/framewin.h>
#include <owl/dialog.h>
#pragma hdrstop
#include "InOut.hpp"
#include "Test.rh"

class TTestdlg : public TDialog{
  public:
    TTestdlg(TWindow* parent, TResId resId);
    ~TTestdlg(){delete Kolektor;}

  protected:
    void SetupWindow(){
      TWindow::SetupWindow();
      CmdResetVBB();
      CmdResetVCC();
    }
    void CmdOk(){Parent->CloseWindow();}
    void SendVBBData(unsigned char Data){
      Kolektor->SendVBBData(Data);
      wsprintf(Buffer, "%02X", Data);
      ::SetWindowText(GetDlgItem(IDC_VBBDATA), Buffer);
    }
    void SendVCCData(unsigned char Data){
      Kolektor->SendVCCData(Data);
      wsprintf(Buffer, "%02X", Data);
      ::SetWindowText(GetDlgItem(IDC_VCCDATA), Buffer);
    }
    void CmdVBBInc(){
      if(VBB == 0xff) return;
      else{
        VBB++;
        SendVBBData(VBB);
      }
    }
    void CmdVBBDec(){
      if(VBB == 0x00) return;
      else{
        VBB--;
        SendVBBData(VBB);
      }
    }
    void CmdResetVBB(){
      SendVBBData(0x00);
      VBB = 0;
    }
    void CmdMaxVBB(){
      SendVBBData(0xff);
      VBB = 0xff;
    }
    void CmdVCCInc(){
      if(VCC == 0xff) return;
      else{
        VCC++;
        SendVCCData(VCC);
      }
    }
    void CmdVCCDec(){
      if(VCC == 0x00) return;
      else{
        VCC--;
        SendVCCData(VCC);
      }
    }
    void CmdResetVCC(){
```

```
        SendVCCData(0x00);
        VCC = 0;
      }
      void CmdMaxVCC(){
        SendVCCData(0xff);
        VCC = 0xff;
      }

    private :
      TKolektor* Kolektor;
      unsigned char VBB;
      unsigned char VCC;
      char Buffer[10];

    DECLARE_RESPONSE_TABLE(TTestdlg);
};

DEFINE_RESPONSE_TABLE1(TTestdlg, TDialog)
    EV_COMMAND(IDOK, CmdOk),
    EV_COMMAND(IDC_VBBDEC, CmdVBBDec),
    EV_COMMAND(IDC_VCCDEC, CmdVCCDec),
    EV_COMMAND(IDC_VBBINC, CmdVBBInc),
    EV_COMMAND(IDC_VCCINC, CmdVCCInc),
    EV_COMMAND(IDC_ResetVBB, CmdResetVBB),
    EV_COMMAND(IDC_ResetVCC, CmdResetVCC),
    EV_COMMAND(IDC_MaxVBB, CmdMaxVBB),
    EV_COMMAND(IDC_MaxVCC, CmdMaxVCC),
END_RESPONSE_TABLE;

TTestdlg::TTestdlg(TWindow* parent, TResId resId)
    :TDialog(parent, resId){
  VBB = 0;
  VCC = 0;
  Kolektor = new TKolektor(0x378, 0x379, 0x37a);
  Kolektor->OutData(0x37a,0x08,true);     // CLK-Off
  Kolektor->OutData(0x37a,0x00,false);    // CLK_On
}

class TMainWnd : public TFrameWindow{
  public :
    TMainWnd(TWindow* parent, const char far* text, TWindow* ch, bool sc);
};

TMainWnd::TMainWnd(TWindow* parent, const char far* text, TWindow* ch, bool sc)
    :TFrameWindow(parent, text, ch, sc){
  Attr.Style &= WS_DLGFRAME;
  Attr.X = 200;
  Attr.Y = 200;
}

class TMyApp : public TApplication{
  public :
    TMyApp():TApplication(){};
    void InitMainWindow();
};

void TMyApp::InitMainWindow(){
  TFrameWindow* MainWnd = new TMainWnd(0, "TestDAC",
                                       new TTestdlg(0, TESTDLGDAC), true);
  SetMainWindow(MainWnd);
}

int OwlMain(int, char**){
  return TMyApp().Run();
}
```

```
#include <owl/applicat.h>
#include <owl/framewin.h>
#include <owl/dialog.h>
#pragma hdrstop
#include "InOut.hpp"
#include "Test.rh"

class TTestdlg : public TDialog{
  public :
    TTestdlg(TWindow* parent, TResId resId);
    ~TTestdlg(){delete Kolektor;}

  protected :
    void TTestdlg::SetupWindow(){
      TWindow::SetupWindow();
      CmdResetVCC();
    }
    void TTestdlg::CmdOk(){Parent->CloseWindow();}
    void TTestdlg::SendVCCData(unsigned char Data){
      Kolektor->SendVCCData(Data);
      wsprintf(Buffer, "%02X", Data);
      ::SetWindowText(GetDlgItem(IDC_VCCDATA), Buffer);
    }
    void TTestdlg::CmdBaca(){
      wsprintf(Buffer,"%02X",Kolektor->GetVCEData());
      ::SetWindowText(GetDlgItem(IDC_ADCDATA), Buffer);
    }
    void TTestdlg::CmdVCCInc(){
      if(VCC == 0xff) return;
      else{
        VCC++;
        SendVCCData(VCC);
      }
    }
    void TTestdlg::CmdVCCDec(){
      if(VCC == 0x00) return;
      else{
        VCC--;
        SendVCCData(VCC);
      }
    }
    void TTestdlg::CmdResetVCC(){
      SendVCCData(0x00);
      VCC = 0;
    }
    void TTestdlg::CmdMaxVCC(){
      SendVCCData(0xff);
      VCC = 0xff;
    }

  private :
    TKolektor* Kolektor;
    unsigned char ADC;
    unsigned char VCC;
    char Buffer[10];

  DECLARE_RESPONSE_TABLE(TTestdlg);
};

DEFINE_RESPONSE_TABLE1(TTestdlg, TDialog)
  EV_COMMAND(IDOK, CmdOk),
  EV_COMMAND(IDC_VCCDEC, CmdVCCDec),
  EV_COMMAND(IDC_VCCINC, CmdVCCInc),
  EV_COMMAND(IDC_ResetVCC, CmdResetVCC),
  EV_COMMAND(IDC_BACA, CmdBaca),
  EV_COMMAND(IDC_MaxVCC, CmdMaxVCC),
```

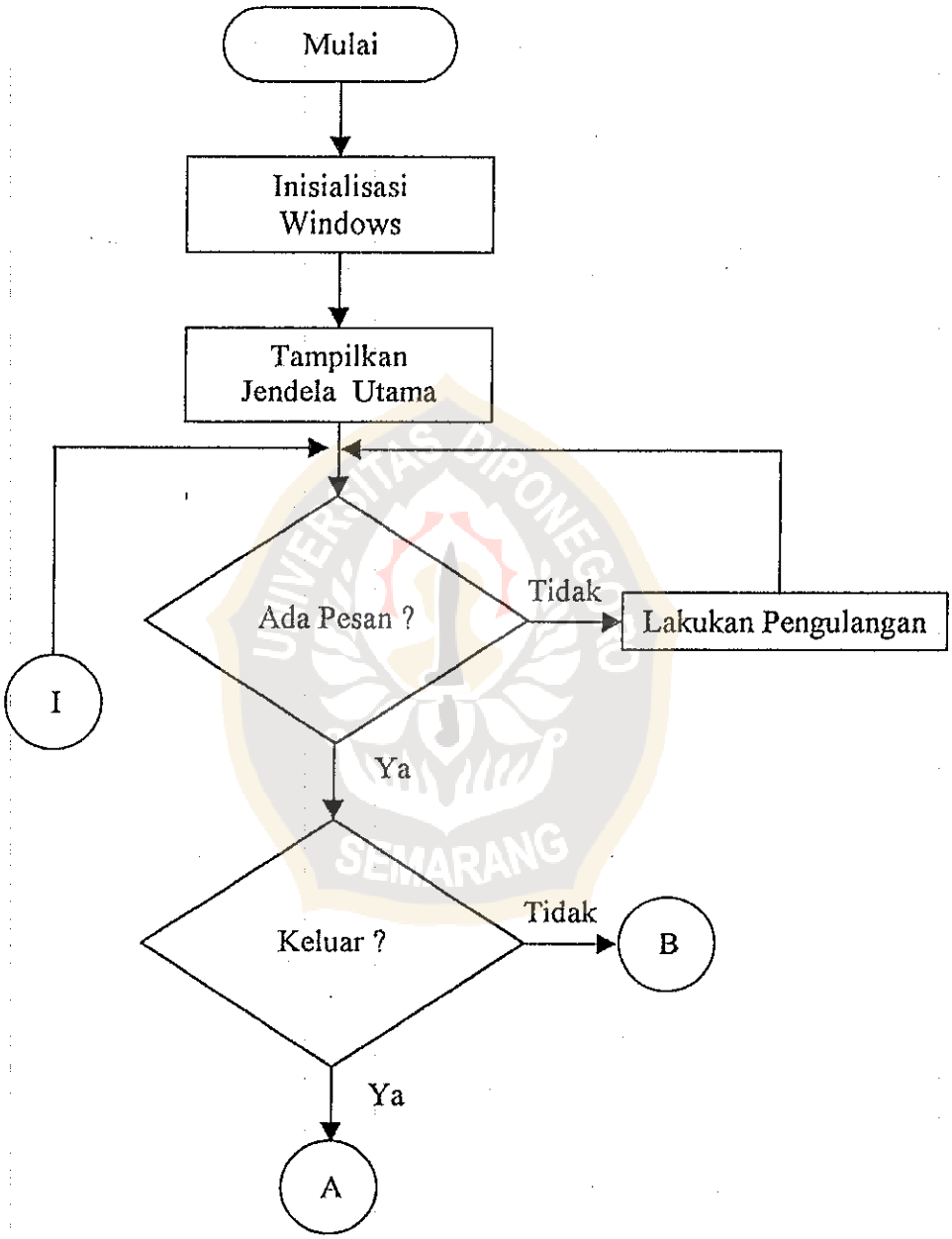
```
END_RESPONSE_TABLE;

TTestdlg::TTestdlg(TWindow* parent, TResId resId)
  :TDialog(parent, resId){
  ADC = 0;
  VCC = 0;
  Kolektor = new TKolektor(0x378, 0x379, 0x37a);
  Kolektor->OutData(0x37a,0x08,true);
  Kolektor->OutData(0x37a,0x00,false);
}

class TMainWnd : public TFrameWindow{
  public:
    TMainWnd(TWindow* parent, const char far* text, TWindow* ch, bool sc);
};

TMainWnd::TMainWnd(TWindow* parent, const char far* text, TWindow* ch, bool sc)
   :TFrameWindow(parent, text, ch, sc){
  Attr.Style &= WS_DLGFRAME;
  Attr.X = 200;
  Attr.Y = 200;
}

class TMyApp : public TApplication{
  public:
    TMyApp():TApplication(){};
    void InitMainWindow(){
      TFrameWindow* MainWnd = new TMainWnd(0, "TestADC",
                                           new TTestdlg(0, TESTDLGADC), true);
      SetMainWindow(MainWnd);
    }

};

int OwlMain(int, char**){
  return TMyApp().Run();
}
```

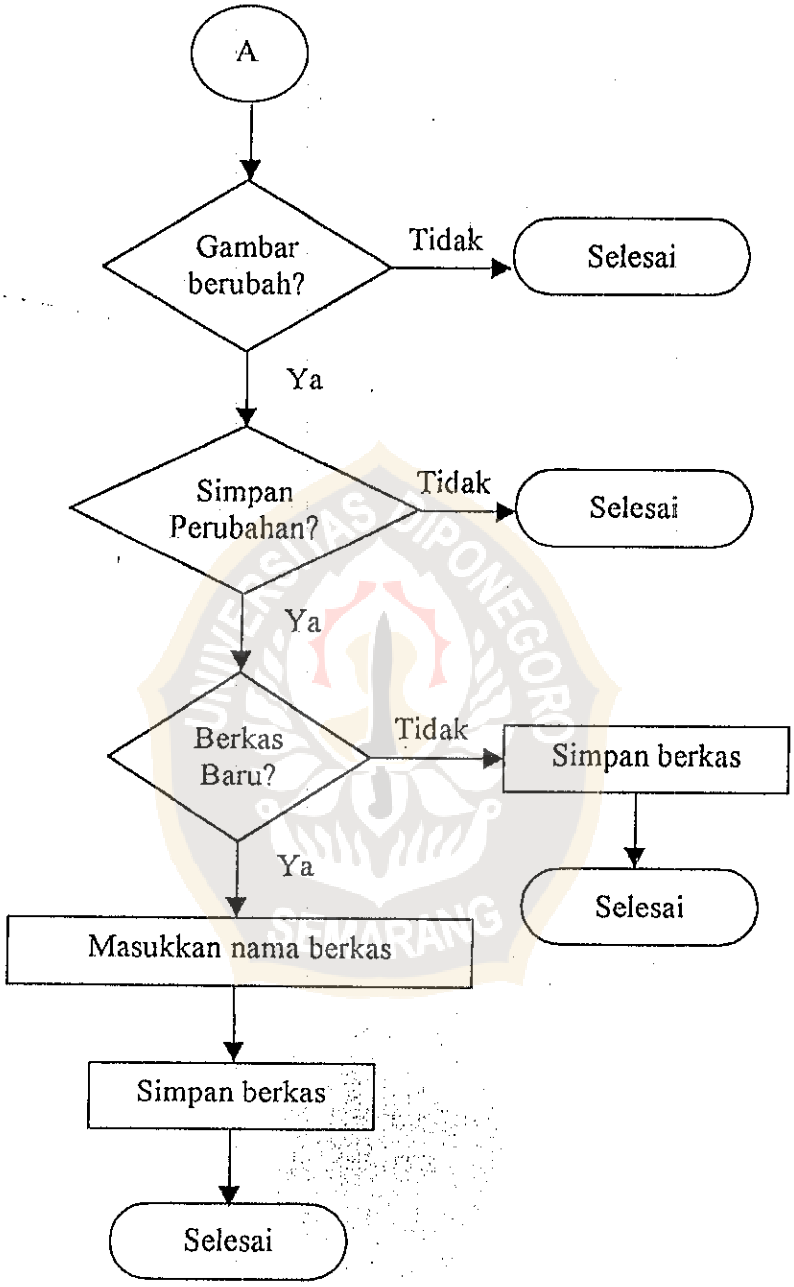
```
//****************************************//
// Test.rc                                //
// Produced by Borland Resource Workshop  //
// Edited by Rahmat Sobari                //
//****************************************//

#include "Test.rh"

TESTDLGADC DIALOG 0, 0, 261, 114
STYLE DS_MODALFRAME | DS_3DLOOK | WS_CHILD | WS_VISIBLE | WS_SYSMENU
FONT 8, "MS Sans Serif"
{
 DEFPUSHBUTTON "&OK", IDOK, 200, 9, 50, 14
 CONTROL "<", IDC_VCCDEC, "button", ButtonStyle, 120, 22, 20, 13
 CONTROL "&Baca", IDC_BACA, "button", ButtonStyle, 28, 39, 48, 14
 CONTROL "Reset", IDC_ResetVCC, "button", ButtonStyle, 120, 39, 48, 14
 CONTROL "Max", IDC_MaxVCC, "button", ButtonStyle, 120, 56, 48, 14
 CONTROL "", IDC_ADCDATA, "static", StaticStyle, 32, 78, 36, 17
 CONTROL "", IDC_VCCDATA, "static", StaticStyle, 124, 78, 36, 17
 CONTROL "ADC", IDC_GROUPBOXADC, "button", GroupBoxStyle, 8, 4, 88, 103
 CONTROL "DAC-C", IDC_GROUPBOXVCC, "button", GroupBoxStyle, 104, 4, 88, 103
 CONTROL ">", IDC_VCCINC, "button", ButtonStyle, 148, 22, 20, 13
}

TESTDLGDAC DIALOG 0, 0, 261, 114
STYLE DS_MODALFRAME | DS_3DLOOK | WS_CHILD | WS_VISIBLE | WS_SYSMENU
FONT 8, "MS Sans Serif"
{
 DEFPUSHBUTTON "&OK", IDOK, 200, 9, 50, 14
 CONTROL "<", IDC_VBBDEC, "button", ButtonStyle, 28, 22, 20, 13
 CONTROL "<", IDC_VCCDEC, "button", ButtonStyle, 120, 22, 20, 13
 CONTROL "Reset", IDC_ResetVBB, "button", ButtonStyle, 28, 39, 48, 14
 CONTROL "Max", IDC_MaxVBB, "button", ButtonStyle, 28, 56, 48, 14
 CONTROL "Reset", IDC_ResetVCC, "button", ButtonStyle, 120, 39, 48, 14
 CONTROL "Max", IDC_MaxVCC, "button", ButtonStyle, 120, 56, 48, 14
 CONTROL "", IDC_VBBDATA, "static", StaticStyle, 32, 78, 36, 17
 CONTROL "", IDC_VCCDATA, "static", StaticStyle, 124, 78, 36, 17
 CONTROL "VBB", IDC_GROUPBOXVBB, "button", GroupBoxStyle, 8, 4, 88, 103
 CONTROL ">", IDC_VBBINC, "button", ButtonStyle, 56, 22, 20, 13
 CONTROL "VCC", IDC_GROUPBOXVCC, "button", GroupBoxStyle, 104, 4, 88, 103
 CONTROL ">", IDC_VCCINC, "button", ButtonStyle, 148, 22, 20, 13
}
```

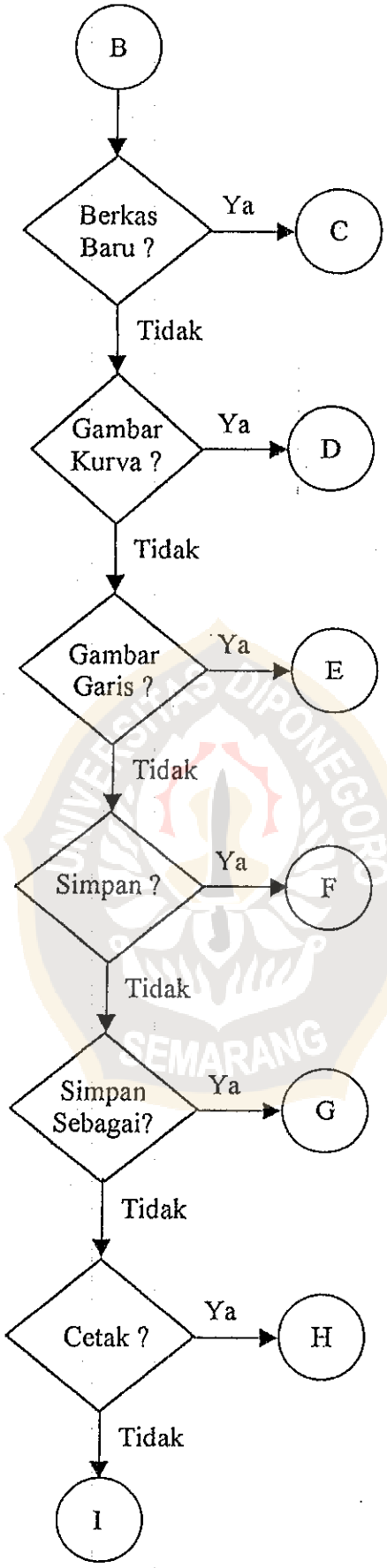
```
#define DialogStyle DS_3DLOOK|WS_POPUP|WS_VISIBLE|WS_CAPTION|WS_SYSMENU
#define ButtonStyle BS_PUSHBUTTON|BS_CENTER|WS_CHILD|WS_VISIBLE|WS_TABSTOP
#define StaticStyle SS_LEFT|WS_CHILD|WS_VISIBLE|WS_BORDER
#define GroupBoxStyle BS_GROUPBOX|WS_CHILD|WS_VISIBLE|WS_GROUP

#define TESTDLGADC              100
#define TESTDLGDAC              200
#define IDC_PASSWIN             300
#define IDC_VCCDEC              301
#define IDC_ResetVCC            302
#define IDC_BACA                303
#define IDC_MaxVCC              304
#define IDC_VCCINC              305
#define IDC_ADCDATA             306
#define IDC_VCCDATA             307
#define IDC_GROUPBOXADC         308
#define IDC_GROUPBOXVCC         309
#define IDC_VBBDEC              310
#define IDC_ResetVBB            311
#define IDC_MaxVBB              312
#define IDC_CLK                 313
#define IDC_VBBINC              314
#define IDC_VBBDATA             315
#define IDC_GROUPBOXVBB         316
```

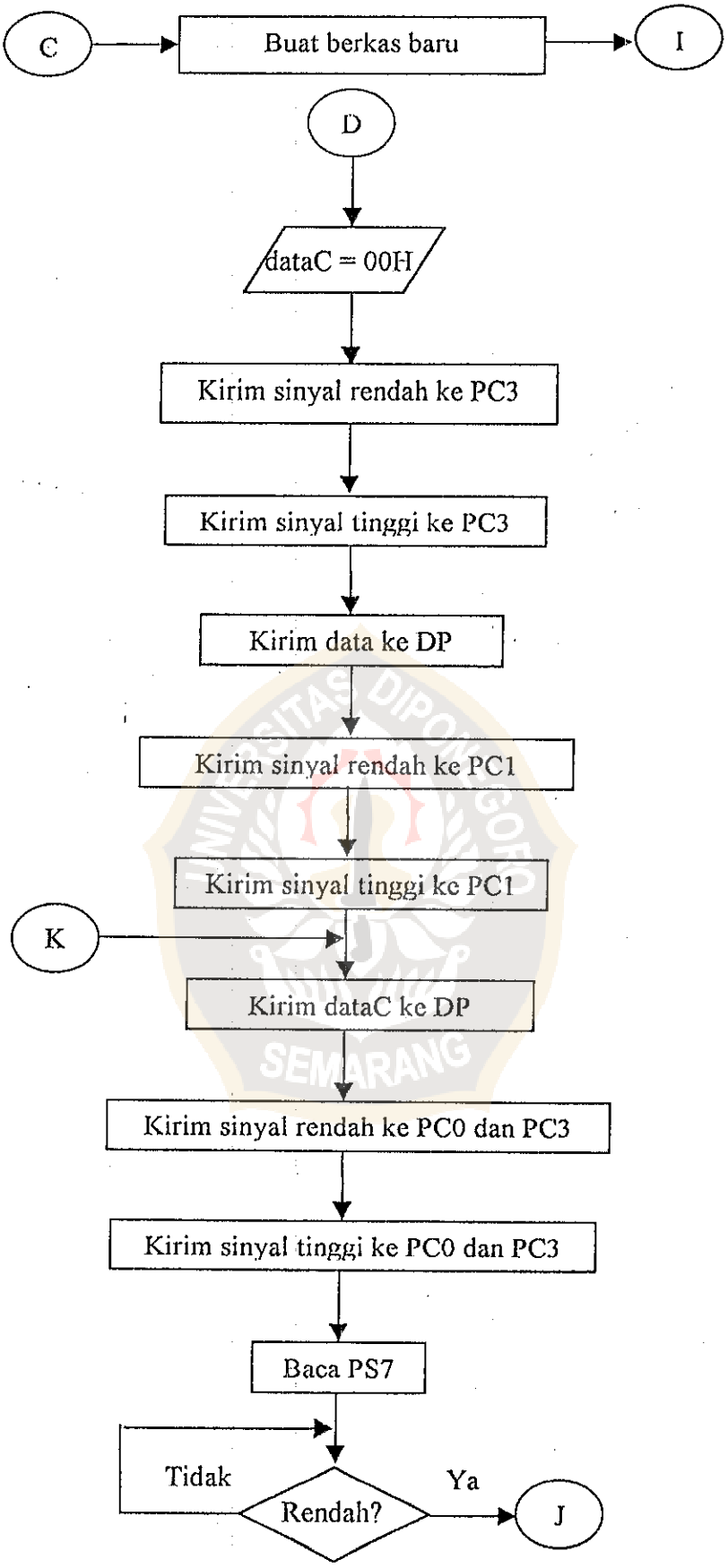
Lampiran II

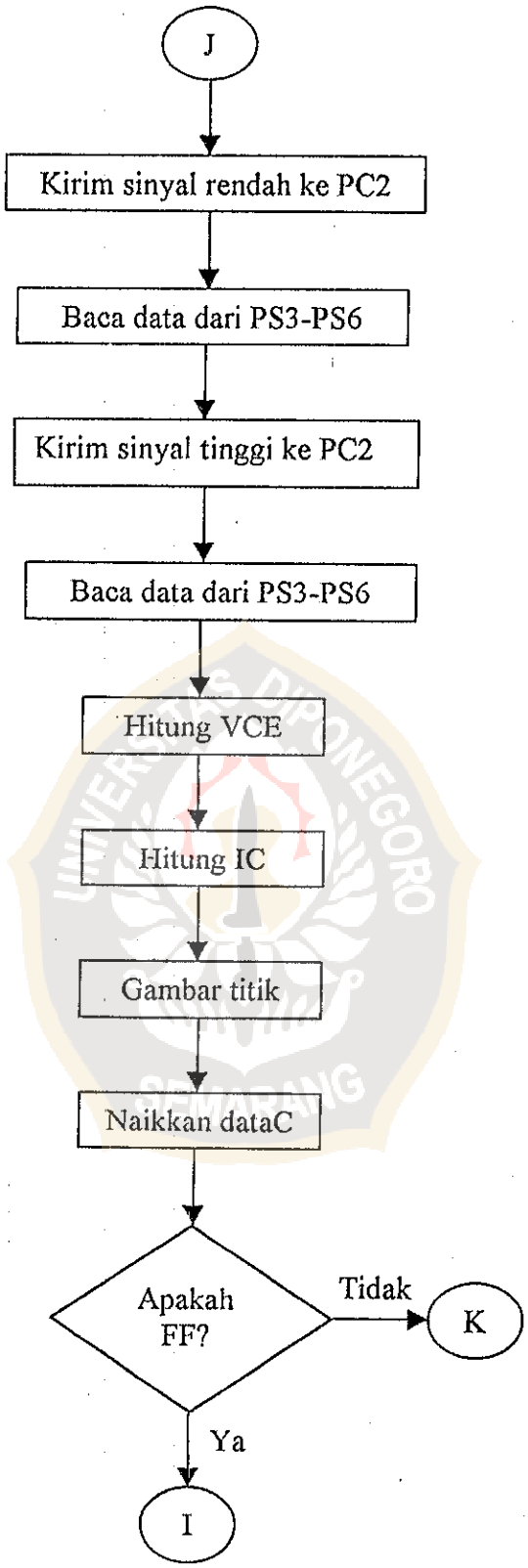
Flow Chart

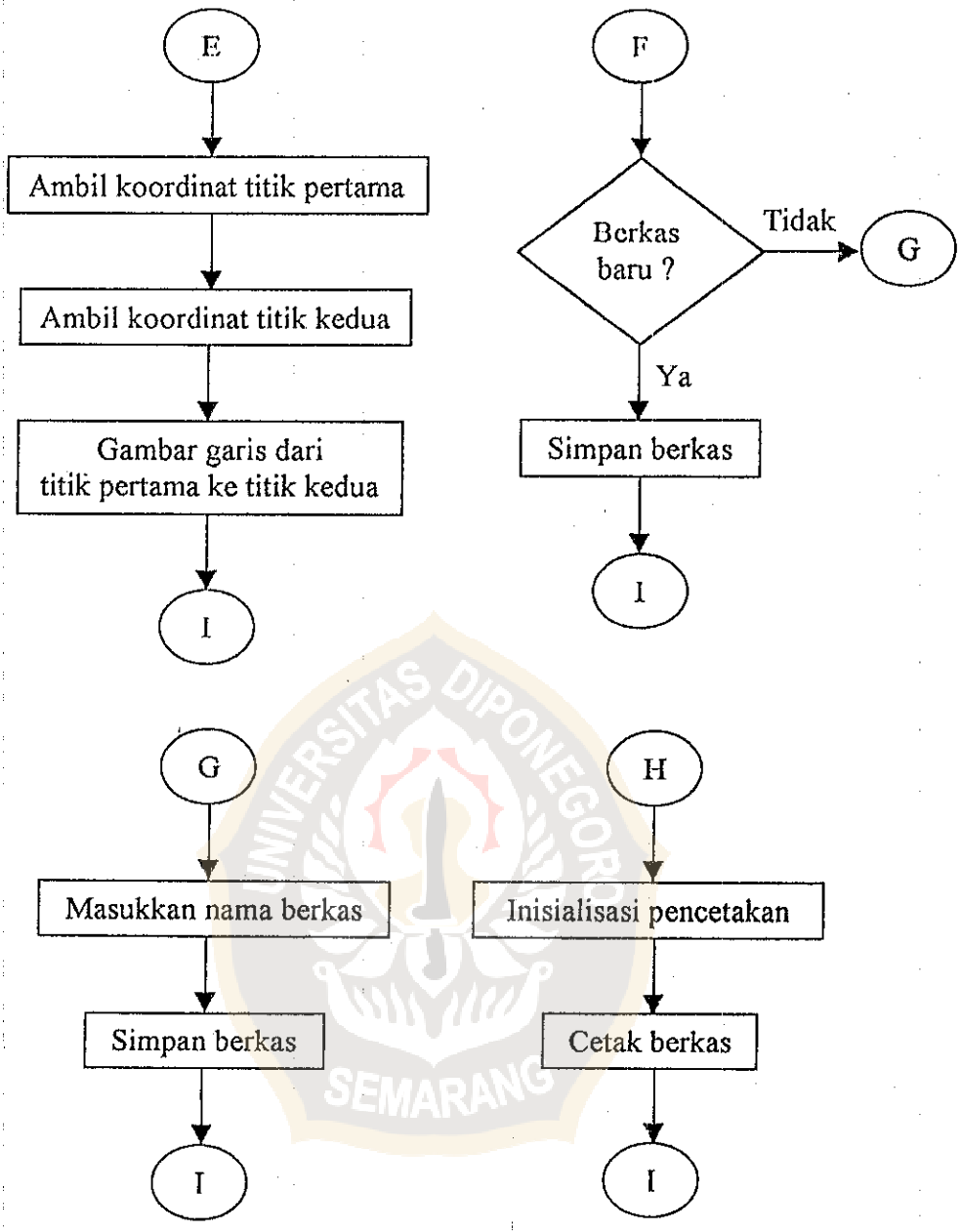












Lampiran III

Perhitungan Ralat dan Regresi Linear

I. Perhitungan ralat hasil pengukuran

Nilai rerata dan ralat yang terdapat pada semua tabel (kecuali perhitungan regresi linear) hasil pengukuran dihitung menggunakan persamaan berikut:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \text{ dan } \sigma_x{}^2 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n(n-1)}$$

Sebagai contoh, pada Tabel 4.2 untuk data terakhir: $Y_1 = 4{,}98$, $Y_2 = 5{,}00$, $Y_3 = 4{,}98$, didapat:

$$\bar{y} = \frac{4{,}98 + 5{,}00 + 4{,}98}{3} = \frac{14{,}96}{3} = 4{,}99$$

$$\sigma_y{}^2 = \frac{(4{,}98 - 4{,}99)^2 + (5{,}00 - 4{,}99)^2 + (4{,}98 - 4{,}99)^2}{3(3-1)} = \frac{3.10^{-4}}{6} = 5.10^{-5}$$

$\sigma_y = \sqrt{5.10^{-5}} = 0{,}007$, dibulatkan sampai dua angka desimal menjadi 0,01, sehingga nilai akhir: $4{,}99 \pm 0{,}01$.

II. Perhitungan persamaan regresi linear dan variansinya

$$\hat{Y} = aX + b$$

dengan $$a = \frac{n\Sigma X_i Y_i - (\Sigma X_i)(\Sigma Y_i)}{n\Sigma X_i^2 - (\Sigma X_i)^2}$$

$$b = \frac{(\Sigma Y_i)(\Sigma X_i^2) - (\Sigma X_i)(\Sigma X_i Y_i)}{n\Sigma X_i^2 - (\Sigma X_i)^2}$$

$$\sigma_y^{\,2} = \frac{\Sigma\left(Y_i - \hat{Y}_i\right)^2}{n-2}$$

1. Regresi linear DAC-B

Data-data berikut ini didapat dari Tabel 4.2:

X	Y_1	Y_2	Y_3	$\bar{Y}$
0	0,00	0,00	0,00	0,00
32	0,65	0,63	0,61	0,63
64	1,27	1,25	1,25	1,26
96	1,88	1,86	1,90	1,88
128	2,53	2,50	2,51	2,51
160	3,14	3,12	3,16	3,14
192	3,74	3,76	3,78	3,76
224	4,38	4,37	4,39	4,38
255	4,98	5,00	4,98	4,99

$\Sigma X_i = 1151 \qquad \Sigma Y_i = 22,55 \qquad \Sigma X_i Y_i = 4080,45$

$\Sigma X_i^2 = 208385 \qquad (\Sigma X_i)^2 = 1324801$

$$a = \frac{9.4080,45 - 1151.22,55}{9.208385 - 1324801} = 0,0196$$

$$b = \frac{22,55.208385 - 1151.4080,45}{9.208385 - 1324801} = 0,0045$$

$$\hat{Y} = 0,0196X + 0,0045$$

$$\Sigma(Y_i - \hat{Y}_i)^2 = 0,0003$$

$$\sigma^2 = \frac{0,0003}{9-2} = \frac{0,0003}{7} = 0,000043 \longrightarrow \sigma = 0,0066$$

2. Regresi linear DAC-C

Data-data berikut ini didapat dari Tabel 4.3:

X	Y_1	Y_2	Y_3	$\bar{Y}$
0	0,00	0,00	0,00	0,00
32	0,65	0,63	0,62	0,63
64	1,25	1,27	1,23	1,25
96	1,86	1,88	1,90	1,88
128	2,52	2,49	2,51	2,51
160	3,14	3,12	3,15	3,14
192	3,74	3,76	3,78	3,76
224	4,39	4,37	4,39	4,38
255	5,00	5,00	4,98	4,99

$$\Sigma X_i = 1151 \qquad \Sigma Y_i = 22{,}54 \qquad \Sigma X_i Y_i = 4079{,}81$$

$$\Sigma X_i^2 = 208385 \qquad (\Sigma X_i)^2 = 1324801$$

$$a = \frac{9.4079{,}81 - 1151.22{,}54}{9.208385 - 1324801} = 0{,}0196$$

$$b = \frac{22{,}54.208385 - 1151.4079{,}81}{9.208385 - 1324801} = 0{,}0021$$

$$Y = 0{,}0196X + 0{,}0021$$

$$\Sigma(Y_i - \hat{Y}_i)^2 = 0{,}0002$$

$$\sigma^2 = \frac{0{,}0002}{9-2} = \frac{0{,}0002}{7} = 0{,}000029 \longrightarrow \sigma = 0{,}0054$$

3. Regresi linear ADC

X	Y_1	Y_2	Y_3	Y
0,00	0	0	0	0
0,63	32	32	34	33
1,25	63	64	67	65
1,88	96	98	98	97
2,51	131	128	127	129
3,14	161	158	162	160
3,76	190	192	193	192
4,38	224	222	226	224
4,99	255	255	253	254

$$\Sigma X_i = 22,54 \qquad \Sigma Y_i = 1154,00 \qquad \Sigma X_i Y_i = 4081,09$$

$$\Sigma X_i^2 = 79,88 \qquad (\Sigma X_i)^2 = 508,05$$

$$a = \frac{9.4081,09 - 22,54.1154,00}{9.79,88 - 508,05} = 50,8$$

$$b = \frac{1154,00.79,88 - 22,54.4081,09}{9.79,88 - 508,05} = 1,0$$

$$\hat{Y} = 50,8X + 1,0$$

$$\Sigma(Y_i - \hat{Y}_i)^2 = 4,00$$

$$\sigma^2 = \frac{4,00}{9-2} = \frac{4,00}{7} = 0,57 \longrightarrow \sigma = 0,75$$