LAMPIRAN A

GAMBAR RANGKAIAN PERANGKAT KERAS DAN FOTO ALAT

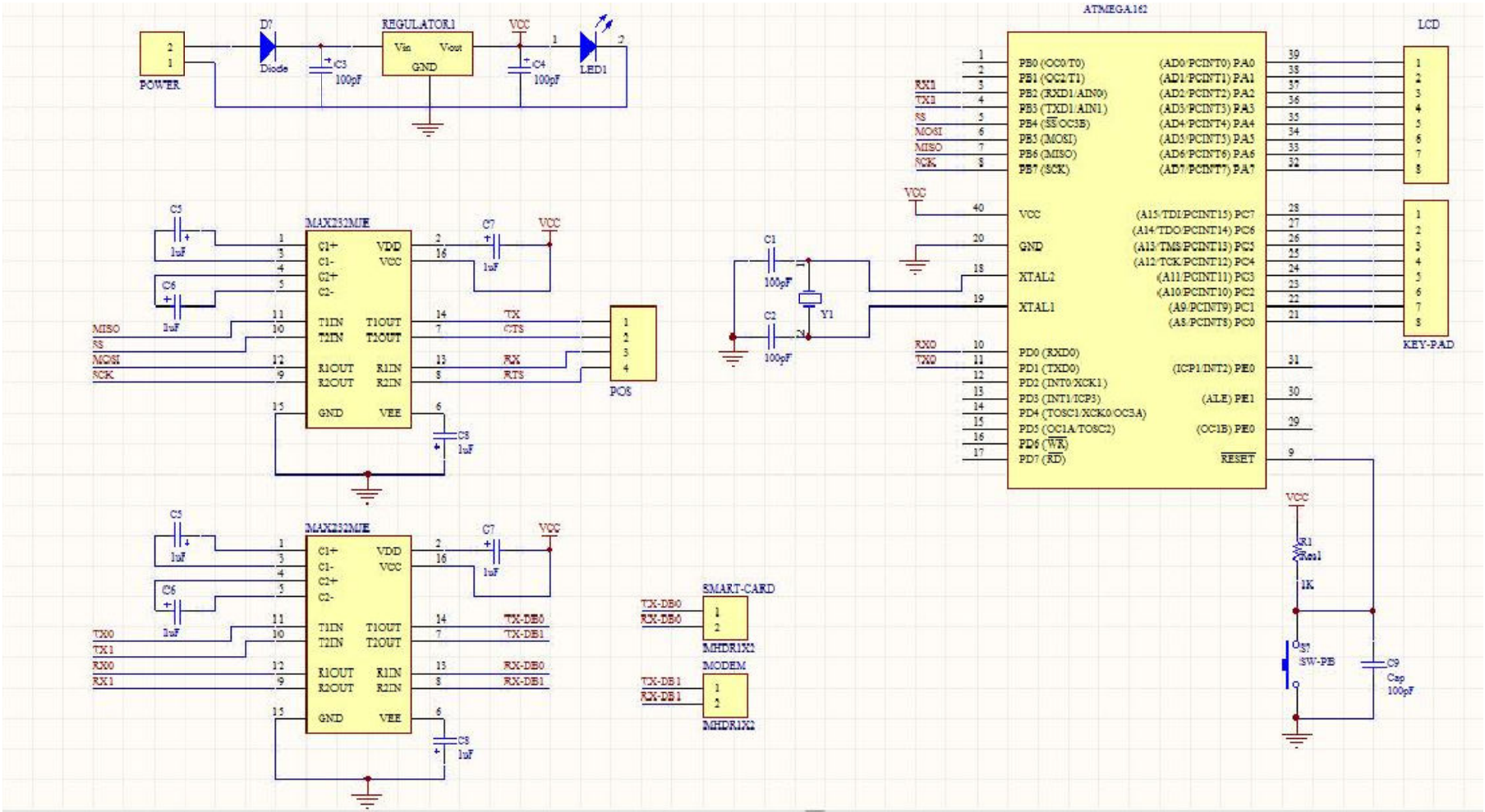
POWER
D7
Diode
C3
100pF
REGULATOR1
Vin
Vout
GND
VCC
C4
100pF
LED1
MAX232MJE
C5
1uF
C6
1uF
C7
1uF
C8
1uF
C1+
C1-
C2+
C2-
VDD
VCC
GND
VEE
T1IN
T2IN
T1OUT
T2OUT
R1OUT
R2OUT
R1IN
R2IN
MISO
SS
MOSI
SCK
TX
CTS
RX
RTS
PC6
TX0
TX1
RX0
RX1
TX-DB0
TX-DB1
RX-DB0
RX-DB1
SMART-CARD
MHDR1X2
MODEM
ATMEGA162
PB0 (OC0/T0)
PB1 (OC2/T1)
PB2 (RXD1/AIN0)
PB3 (TXD1/AIN1)
PB4 (SS/OC3B)
PB5 (MOSI)
PB6 (MISO)
PB7 (SCK)
VCC
GND
XTAL2
XTAL1
PD0 (RXD0)
PD1 (TXD0)
PD2 (INT0/XCK1)
PD3 (INT1/ICP3)
PD4 (TOSC1/XCK0/OC3A)
PD5 (OC1A/TOSC2)
PD6 (WR)
PD7 (RD)
(AD0/PCINT0) PA0
(AD1/PCINT1) PA1
(AD2/PCINT2) PA2
(AD3/PCINT3) PA3
(AD4/PCINT4) PA4
(AD5/PCINT5) PA5
(AD6/PCINT6) PA6
(AD7/PCINT7) PA7
(A15/TDI/PCINT15) PC7
(A14/TDO/PCINT14) PC6
(A13/TMS/PCINT13) PC5
(A12/TCK/PCINT12) PC4
(A11/PCINT11) PC3
(A10/PCINT10) PC2
(A9/PCINT9) PC1
(A8/PCINT8) PC0
(ICP1/INT2) PE0
(ALE) PE1
(OC1B) PE0
RESET
C1
100pF
C2
100pF
Y1
RXD
TXD
LCD
KEY-PAD
R1
Res1
1K
SW-PB
C9
Cap
100pF

FOTO ALAT

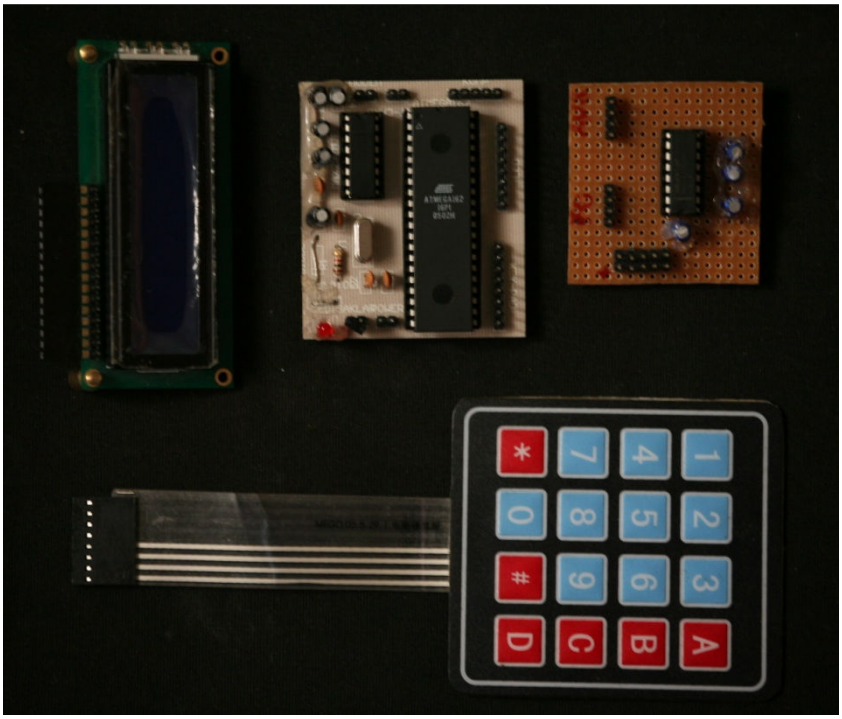

Foto Rangkaian Sistem Minimum ATMega 162

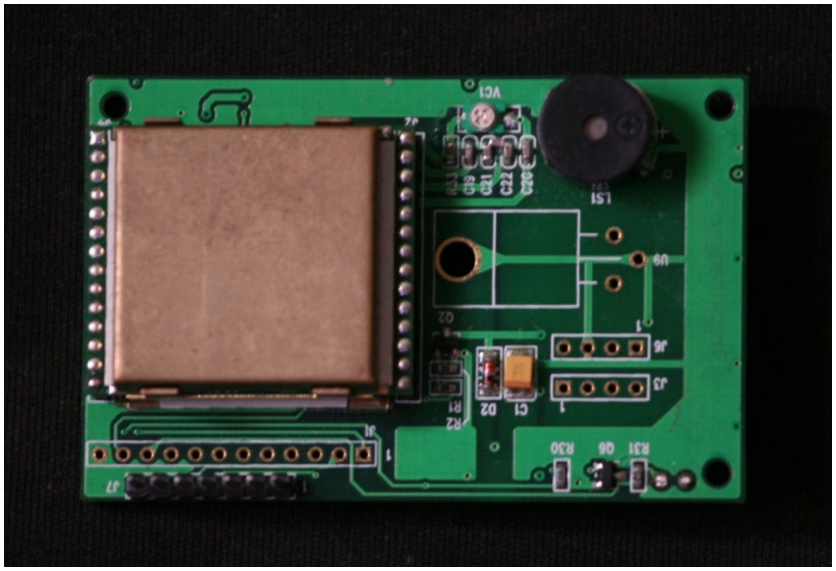

Foto *Smartcard Reader* ACM-120

Foto *Smartcard* MIFARE 4KByte

Foto Modem GPRS

Foto Sistem Secara Keseluruhan

PERANGKAT LUNAK

KODE SUMBER

- **POS (*Point Of Sales*)**

```
// POSDlg.cpp : implementation file
//

#include "stdafx.h"
#include "POS.h"
#include "POSDlg.h"
#include "PProduk.h"
#include "PTelepon.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
    CAboutDlg();

// Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    //{{AFX_MSG(CAboutDlg)
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
            // No message handlers
    //}}AFX_MSG_MAP
```

```
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CPOSDlg dialog

CPOSDlg::CPOSDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CPOSDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CPOSDlg)
	m_InfoStatic = _T(" ");
	//}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

	m_bBtnConnect = FALSE;

}

void CPOSDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CPOSDlg)
	DDX_Control(pDX, IDC_CMB_PORT, m_CmbPort);
	DDX_Control(pDX, IDC_BTN_CONNECT, m_BtnConnect);
	DDX_Text(pDX, IDC_STAT_INFO, m_InfoStatic);
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CPOSDlg, CDialog)
	//{{AFX_MSG_MAP(CPOSDlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_BTN_CONNECT, OnBtnConnect)
	ON_BN_CLICKED(IDC_BTN_PPRODUK, OnBtnPproduk)
	ON_BN_CLICKED(IDC_BTN_PTELP, OnBtnPtelp)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CPOSDlg message handlers

BOOL CPOSDlg::OnInitDialog()
{
	CDialog::OnInitDialog();

	// Add "About..." menu item to system menu.

	// IDM_ABOUTBOX must be in the system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
		}
	}
```

```
	// Set the icon for this dialog.  The framework does this automatically
	//  when the application's main window is not a dialog
	SetIcon(m_hIcon, TRUE);			// Set big icon
	SetIcon(m_hIcon, FALSE);		// Set small icon

	// TODO: Add extra initialization here

	return TRUE;  // return TRUE  unless you set the focus to a control
}

void CPOSDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)
	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	}
	else
	{
		CDialog::OnSysCommand(nID, lParam);
	}
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CPOSDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // device context for painting

		SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

		// Center icon in client rectangle
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;

		// Draw the icon
		dc.DrawIcon(x, y, m_hIcon);
	}
	else
	{
		CDialog::OnPaint();
	}
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CPOSDlg::OnQueryDragIcon()
{
	return (HCURSOR) m_hIcon;
}

void CPOSDlg::OnBtnConnect()
{
	// TODO: Add your control notification handler code here
	if(m_bBtnConnect == FALSE)
	{
```

```
		// Open COM port
		CString strPilih;
		int indexPilih = m_CmbPort.GetCurSel();

		if(indexPilih != CB_ERR)
		{
			m_CmbPort.GetLBText(indexPilih, strPilih);
			UpdateData(FALSE);
		}

		t_errcode Err = OpenComPort(hComm, strPilih);
		if(Err == SUCCESS)
		{
			m_InfoStatic.Format("Connecting to %s",strPilih);
			m_bBtnConnect = TRUE;
			m_BtnConnect.SetWindowText("&Disconnect");
		}
		else
		{
			MessageBox("Serial Port Opening Failure");
		}
	}
	else
	{
		m_InfoStatic = "COM Port Disonnected";
		m_bBtnConnect = FALSE;
		m_BtnConnect.SetWindowText("&Connect");
	}
	UpdateData(FALSE);
}

t_errcode CPOSDlg::getPortName(CString sPort, LPCTSTR &portName)
{
	t_errcode retVal = FAILURE;
	CString str1, str2;

	str1 = str2 = sPort;
	str1 = str1.Left(3);
	str2 = str2.Right(1);

	if(str1.Compare("COM") == 0)
	{
		if(str2.Compare("1") == 0)
		{
			portName = "COM1:";
			retVal = SUCCESS;
		}
		else if(str2.Compare("2") == 0)
		{
			portName = "COM2:";
			retVal = SUCCESS;
		}
		else
			retVal = FAILURE;
	}
	else
	{
		retVal = FAILURE;
	}
	return retVal;
}

t_errcode CPOSDlg::OpenComPort(HANDLE phPort, CString sPort)
{
```

```
	t_errcode retVal = SUCCESS;
	t_errcode errCode;
	LPCTSTR str;

	errCode = getPortName(sPort, str);
	if(errCode == FAILURE)
		return FAILURE;

	phPort = CreateFile(str,
						GENERIC_READ | GENERIC_WRITE,
						0,
						NULL,
						OPEN_EXISTING,
						0,
						NULL);
	if(phPort == INVALID_HANDLE_VALUE)
		retVal = SUCCESS;
	return retVal;
}

void CPOSDlg::CloseComPort(HANDLE phPort)
{
	CloseHandle(&phPort);
}

BOOL CPOSDlg::WriteComPort(char *lpBuf, DWORD dwToWrite)
{
	BOOL fRes;

	OVERLAPPED osWrite = {0};
	DWORD dwWritten;

	//Writes OVERLAPPED structure hEvent
	osWrite.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
	if(osWrite.hEvent == NULL)
		return FALSE;		//error create overlapped event handle.

	// Issue write to port
	if(!WriteFile(hComm, lpBuf, dwToWrite, &dwWritten, &osWrite))
	{
		if(GetLastError() != ERROR_IO_PENDING)
			fRes = FALSE; //Write File failed
		else
		{
			if(!GetOverlappedResult(hComm, &osWrite, &dwWritten,TRUE))
				fRes = FALSE;
			else
				fRes = TRUE;		// write operation completed successfully
		}
	}
	else
		fRes = TRUE;		// WriteFile completed immediately

	CloseHandle(osWrite.hEvent);
	return fRes;
}

void CPOSDlg::OnBtnPproduk()
{
	// TODO: Add your control notification handler code here
	if(m_bBtnConnect == FALSE)
	{
		m_InfoStatic = "COM Port connection Error!";
	}
```

```
    else
    {
            m_InfoStatic = "Waiting...";

            CPProduk dlg;
            dlg.DoModal();

            if(dlg.m_dTombol == IDOK)
            {
                    // waiting confirmation from uController
                    m_InfoStatic = "Transaksi Pembelian Produk - OK";
            }
            else
            {
                    m_InfoStatic = "Transaksi Pembelian Produk - ABORT";
            }
    }
    UpdateData(FALSE);
}

void CPOSDlg::OnBtnPtelp()
{
    // TODO: Add your control notification handler code here
    if(m_bBtnConnect == FALSE)
    {
            m_InfoStatic = "COM Port connection Error!";
    }
    else
    {
            m_InfoStatic = "Waiting...";

            CPTelepon dlg;
            dlg.DoModal();

            if(dlg.m_dTombol == IDOK)
            {
                    // waiting confirmation from uController
                    m_InfoStatic = "Transaksi Pembayaran Telepon - OK";
            }
            else
                    m_InfoStatic = "Transaksi Pembayaran Telepon - ABORT";
    }
    UpdateData(FALSE);
}
```

// POS.h : main header file for the POS application
//

```
#if !defined(AFX_POS_H__094B8891_7141_446E_AA58_7498708B488F__INCLUDED_)
#define AFX_POS_H__094B8891_7141_446E_AA58_7498708B488F__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#ifndef __AFXWIN_H__
    #error include 'stdafx.h' before including this file for PCH
#endif

#include "resource.h"                   // main symbols

/////////////////////////////////////////////////////////////////////////////
// CPOSApp:
// See POS.cpp for the implementation of this class
```

```
//

class CPOSApp : public CWinApp
{
public:
	CPOSApp();

// Overrides
	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CPOSApp)
	public:
	virtual BOOL InitInstance();
	//}}AFX_VIRTUAL

// Implementation

	//{{AFX_MSG(CPOSApp)
			// NOTE - the ClassWizard will add and remove member functions here.
			//    DO NOT EDIT what you see in these blocks of generated code !
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined(AFX_POS_H__094B8891_7141_446E_AA58_7498708B488F__INCLUDED_)
```

// POSDlg.h : header file

```
//

#if !defined(AFX_POSDLG_H__4999BCE3_AEAF_4176_BE7E_D1CB4F433376__INCLUDED_)
#define AFX_POSDLG_H__4999BCE3_AEAF_4176_BE7E_D1CB4F433376__INCLUDED_

#include "PProduk.h"	// Added by ClassView
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#define SUCCESS			0
#define FAILURE			-1

typedef signed int t_errcode;

/////////////////////////////////////////////////////////////////////////////
// CPOSDlg dialog
class CPProduk;
class CPTelepon;

class CPOSDlg : public CDialog
{
// Construction
public:
	CPOSDlg(CWnd* pParent = NULL);	// standard constructor

// Dialog Data
	//{{AFX_DATA(CPOSDlg)
	enum { IDD = IDD_POS_DIALOG };
	CComboBox		m_CmbPort;
	CButton	m_BtnConnect;
	CString	m_InfoStatic;
```

```
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CPOSDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);       // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    void HitungSubTotal();
    BOOL WriteComPort(char *lpBuf, DWORD dwToWrite);
    void CloseComPort(HANDLE phPort);
    t_errcode getPortName(CString sPort, LPCTSTR &portName);
    t_errcode OpenComPort(HANDLE phPort, CString sPort);
    HANDLE *hComm;
    bool m_bBtnConnect;
    HICON m_hIcon;

    // Generated message map functions
    //{{AFX_MSG(CPOSDlg)
    virtual BOOL OnInitDialog();
    afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    afx_msg void OnBtnConnect();
    afx_msg void OnBtnPproduk();
    afx_msg void OnBtnPtelp();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif //
!defined(AFX_POSDLG_H__4999BCE3_AEAF_4176_BE7E_D1CB4F433376__INCLUDED_)


#if !defined(AFX_PPRODUK_H__FF64CA61_F92D_4105_858E_DA64CE9E72C6__INCLUDED_)
#define AFX_PPRODUK_H__FF64CA61_F92D_4105_858E_DA64CE9E72C6__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// PProduk.h : header file
//

/////////////////////////////////////////////////////////////////////////////
// CPProduk dialog

class CPProduk : public CDialog
{
// Construction
public:
    int m_dTombol;
    CPProduk(CWnd* pParent = NULL);   // standard constructor

// Dialog Data
    //{{AFX_DATA(CPProduk)
    enum { IDD = IDD_DLG_PPRODUK };
    CComboBox       m_CmbPrd4;
    CComboBox       m_CmbPrd3;
    CComboBox       m_CmbPrd2;
```

```
    CComboBox       m_CmbPrd1;
    CComboBox       m_CmbPrd0;
    CString   m_EdtDis0;
    CString   m_EdtDis1;
    CString   m_EdtDis2;
    CString   m_EdtDis3;
    CString   m_EdtDis4;
    CString   m_EdtHrg0;
    CString   m_EdtHrg1;
    CString   m_EdtHrg2;
    CString   m_EdtHrg3;
    CString   m_EdtHrg4;
    UINT      m_EdtJlh2;
    UINT      m_EdtJlh0;
    UINT      m_EdtJlh1;
    UINT      m_EdtJlh3;
    UINT      m_EdtJlh4;
    CString   m_TotalInfo;
    //}}AFX_DATA


// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CPProduk)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    void HitungTotal();

    // Generated message map functions
    //{{AFX_MSG(CPProduk)
            virtual BOOL OnInitDialog();
    afx_msg void OnCloseupCmbNmprd0();
    afx_msg void OnCloseupCmbNmprd1();
    afx_msg void OnCloseupCmbNmprd2();
    afx_msg void OnCloseupCmbNmprd3();
    afx_msg void OnCloseupCmbNmprd4();
    afx_msg void OnChangeEdtJlh0();
    afx_msg void OnKillfocusEdtJlh0();
    afx_msg void OnKillfocusEdtJlh1();
    afx_msg void OnKillfocusEdtJlh2();
    afx_msg void OnKillfocusEdtJlh3();
    afx_msg void OnKillfocusEdtJlh4();
    virtual void OnOK();
    virtual void OnCancel();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif //
!defined(AFX_PPRODUK_H__FF64CA61_F92D_4105_858E_DA64CE9E72C6__INCLUDED_)

#if
!defined(AFX_PTELEPON_H__A3FA12C9_5077_49C7_A9C9_B1AFC3DA0F1A__INCLUDED_)
#define AFX_PTELEPON_H__A3FA12C9_5077_49C7_A9C9_B1AFC3DA0F1A__INCLUDED_

#if _MSC_VER > 1000
#pragma once
```

```
#endif // _MSC_VER > 1000
// PTelepon.h : header file
//

/////////////////////////////////////////////////////////////////////////////
// CPTelepon dialog

class CPTelepon : public CDialog
{
// Construction
public:
	int m_dTombol;
	CPTelepon(CWnd* pParent = NULL);   // standard constructor

// Dialog Data
	//{{AFX_DATA(CPTelepon)
	enum { IDD = IDD_DLG_PTLP };
	CString	m_EdtRek;
	//}}AFX_DATA


// Overrides
	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CPTelepon)
	protected:
	virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV support
	//}}AFX_VIRTUAL

// Implementation
protected:

	// Generated message map functions
	//{{AFX_MSG(CPTelepon)
	virtual void OnOK();
	virtual void OnCancel();
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif //
!defined(AFX_PTELEPON_H__A3FA12C9_5077_49C7_A9C9_B1AFC3DA0F1A__INCLUDED_)

// PProduk.cpp : implementation file
//

#include "stdafx.h"
#include "POS.h"
#include "PProduk.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

char *tblHarga[]                    = {"10000","50000","100000","10000","10000"};
char *tblDiscount[]                 = {"0.1", "0.3", "0.2", "0.1", "0.5"};
char *tblNamaProduk[] = {"PRODUK1", "PRODUK2", "PRODUK3", "PRODUK4", "PRODUK5"};
/////////////////////////////////////////////////////////////////////////////
// CPProduk dialog
```

```
CPProduk::CPProduk(CWnd* pParent /*=NULL*/)
	: CDialog(CPProduk::IDD, pParent)
{
	//{{AFX_DATA_INIT(CPProduk)
	m_EdtDis0 = _T("");
	m_EdtDis1 = _T("");
	m_EdtDis2 = _T("");
	m_EdtDis3 = _T("");
	m_EdtDis4 = _T("");
	m_EdtHrg0 = _T("");
	m_EdtHrg1 = _T("");
	m_EdtHrg2 = _T("");
	m_EdtHrg3 = _T("");
	m_EdtHrg4 = _T("");
	m_EdtJlh2 = 0;
	m_EdtJlh0 = 0;
	m_EdtJlh1 = 0;
	m_EdtJlh3 = 0;
	m_EdtJlh4 = 0;
	m_TotalInfo = _T("");
	//}}AFX_DATA_INIT
}


void CPProduk::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CPProduk)
	DDX_Control(pDX, IDC_CMB_NMPRD4, m_CmbPrd4);
	DDX_Control(pDX, IDC_CMB_NMPRD3, m_CmbPrd3);
	DDX_Control(pDX, IDC_CMB_NMPRD2, m_CmbPrd2);
	DDX_Control(pDX, IDC_CMB_NMPRD1, m_CmbPrd1);
	DDX_Control(pDX, IDC_CMB_NMPRD0, m_CmbPrd0);
	DDX_Text(pDX, IDC_EDT_DSC0, m_EdtDis0);
	DDX_Text(pDX, IDC_EDT_DSC1, m_EdtDis1);
	DDX_Text(pDX, IDC_EDT_DSC2, m_EdtDis2);
	DDX_Text(pDX, IDC_EDT_DSC3, m_EdtDis3);
	DDX_Text(pDX, IDC_EDT_DSC4, m_EdtDis4);
	DDX_Text(pDX, IDC_EDT_HARGA0, m_EdtHrg0);
	DDX_Text(pDX, IDC_EDT_HARGA1, m_EdtHrg1);
	DDX_Text(pDX, IDC_EDT_HARGA2, m_EdtHrg2);
	DDX_Text(pDX, IDC_EDT_HARGA3, m_EdtHrg3);
	DDX_Text(pDX, IDC_EDT_HARGA4, m_EdtHrg4);
	DDX_Text(pDX, IDC_EDT_JLH2, m_EdtJlh2);
	DDX_Text(pDX, IDC_EDT_JLH0, m_EdtJlh0);
	DDX_Text(pDX, IDC_EDT_JLH1, m_EdtJlh1);
	DDX_Text(pDX, IDC_EDT_JLH3, m_EdtJlh3);
	DDX_Text(pDX, IDC_EDT_JLH4, m_EdtJlh4);
	DDX_Text(pDX, IDC_STAT_TOT, m_TotalInfo);
	//}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CPProduk, CDialog)
	//{{AFX_MSG_MAP(CPProduk)
	ON_CBN_CLOSEUP(IDC_CMB_NMPRD0, OnCloseupCmbNmprd0)
	ON_CBN_CLOSEUP(IDC_CMB_NMPRD1, OnCloseupCmbNmprd1)
	ON_CBN_CLOSEUP(IDC_CMB_NMPRD2, OnCloseupCmbNmprd2)
	ON_CBN_CLOSEUP(IDC_CMB_NMPRD3, OnCloseupCmbNmprd3)
	ON_CBN_CLOSEUP(IDC_CMB_NMPRD4, OnCloseupCmbNmprd4)
	ON_EN_KILLFOCUS(IDC_EDT_JLH0, OnKillfocusEdtJlh0)
	ON_EN_KILLFOCUS(IDC_EDT_JLH1, OnKillfocusEdtJlh1)
	ON_EN_KILLFOCUS(IDC_EDT_JLH2, OnKillfocusEdtJlh2)
```

```
    ON_EN_KILLFOCUS(IDC_EDT_JLH3, OnKillfocusEdtJlh3)
    ON_EN_KILLFOCUS(IDC_EDT_JLH4, OnKillfocusEdtJlh4)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

BOOL CPProduk::OnInitDialog()
{
    CDialog::OnInitDialog();
    for(int i=0; i<5; i++)
    {
            m_CmbPrd0.AddString(tblNamaProduk[i]);
            m_CmbPrd1.AddString(tblNamaProduk[i]);
            m_CmbPrd2.AddString(tblNamaProduk[i]);
            m_CmbPrd3.AddString(tblNamaProduk[i]);
            m_CmbPrd4.AddString(tblNamaProduk[i]);
    }
    return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CPProduk message handlers

void CPProduk::OnCloseupCmbNmprd0()
{
    // TODO: Add your control notification handler code here
    int index = m_CmbPrd0.GetCurSel();
    m_EdtHrg0 = tblHarga[index];
    m_EdtDis0 = tblDiscount[index];

    UpdateData(FALSE);
}

void CPProduk::OnCloseupCmbNmprd1()
{
    // TODO: Add your control notification handler code here
    int index = m_CmbPrd1.GetCurSel();
    m_EdtHrg1 = tblHarga[index];
    m_EdtDis1 = tblDiscount[index];

    UpdateData(FALSE);
}

void CPProduk::OnCloseupCmbNmprd2()
{
    // TODO: Add your control notification handler code here
    int index = m_CmbPrd2.GetCurSel();
    m_EdtHrg2 = tblHarga[index];
    m_EdtDis2 = tblDiscount[index];

    UpdateData(FALSE);
}

void CPProduk::OnCloseupCmbNmprd3()
{
    // TODO: Add your control notification handler code here
    int index = m_CmbPrd3.GetCurSel();
    m_EdtHrg3 = tblHarga[index];
    m_EdtDis3 = tblDiscount[index];

    UpdateData(FALSE);
}

void CPProduk::OnCloseupCmbNmprd4()
{
```

```
	// TODO: Add your control notification handler code here
	int index = m_CmbPrd4.GetCurSel();
	m_EdtHrg4 = tblHarga[index];
	m_EdtDis4 = tblDiscount[index];

	UpdateData(FALSE);
}

void CPProduk::HitungTotal()
{
	UpdateData(TRUE);
	double fSubTotal[5] = {0};
	double total = 0;

	fSubTotal[0] = (float)(atoi(m_EdtHrg0) * m_EdtJlh0) * atof(m_EdtDis0);
	fSubTotal[1] = (float)(atoi(m_EdtHrg1) * m_EdtJlh1) * atof(m_EdtDis1);
	fSubTotal[2] = (float)(atoi(m_EdtHrg2) * m_EdtJlh2) * atof(m_EdtDis2);
	fSubTotal[3] = (float)(atoi(m_EdtHrg3) * m_EdtJlh3) * atof(m_EdtDis3);
	fSubTotal[4] = (float)(atoi(m_EdtHrg4) * m_EdtJlh4) * atof(m_EdtDis4);

	for(int i=0 ;i<5; i++)
		total += fSubTotal[i];

	m_TotalInfo.Format("%.2f", total);

	UpdateData(FALSE);
}

void CPProduk::OnKillfocusEdtJlh0()
{
	// TODO: Add your control notification handler code here
	HitungTotal();
}

void CPProduk::OnKillfocusEdtJlh1()
{
	// TODO: Add your control notification handler code here
	HitungTotal();
}

void CPProduk::OnKillfocusEdtJlh2()
{
	// TODO: Add your control notification handler code here
	HitungTotal();
}

void CPProduk::OnKillfocusEdtJlh3()
{
	// TODO: Add your control notification handler code here
	HitungTotal();
}

void CPProduk::OnKillfocusEdtJlh4()
{
	// TODO: Add your control notification handler code here
	HitungTotal();
}

void CPProduk::OnOK()
{
	// TODO: Add extra validation here
	CPProduk::m_dTombol = IDOK;

	//Write Total to COM Port
```

```
    CDialog::OnOK();
}

void CPProduk::OnCancel()
{
    // TODO: Add extra cleanup here
    CPProduk::m_dTombol = IDCANCEL;
    CDialog::OnCancel();
}

// POS.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "POS.h"
#include "POSDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CPOSApp

BEGIN_MESSAGE_MAP(CPOSApp, CWinApp)
    //{{AFX_MSG_MAP(CPOSApp)
            // NOTE - the ClassWizard will add and remove mapping macros here.
            //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CPOSApp construction

CPOSApp::CPOSApp()
{
    // TODO: add construction code here,
    // Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CPOSApp object

CPOSApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CPOSApp initialization

BOOL CPOSApp::InitInstance()
{
    AfxEnableControlContainer();

    // Standard initialization
    // If you are not using these features and wish to reduce the size
    //  of your final executable, you should remove from the following
    //  the specific initialization routines you do not need.

#ifdef _AFXDLL
    Enable3dControls();                     // Call this when using MFC in a shared DLL
#else
```

```
	Enable3dControlsStatic();	// Call this when linking to MFC statically
#endif

	CPOSDlg dlg;
	m_pMainWnd = &dlg;
	int nResponse = dlg.DoModal();
	if (nResponse == IDOK)
	{
		// TODO: Place code here to handle when the dialog is
		//  dismissed with OK
	}
	else if (nResponse == IDCANCEL)
	{
		// TODO: Place code here to handle when the dialog is
		//  dismissed with Cancel
	}

	// Since the dialog has been closed, return FALSE so that we exit the
	//  application, rather than start the application's message pump.
	return FALSE;
}
```

- **Menu Daftar Nasabah**

```
	// MDNDlg.cpp : implementation file
	//

	#include "stdafx.h"
	#include "MDN.h"
	#include "MDNDlg.h"

	#ifdef _DEBUG
	#define new DEBUG_NEW
	#undef THIS_FILE
	static char THIS_FILE[] = __FILE__;
	#endif


	#define LOCALHOST "localhost"
	#define USERNAME  "root"
	#define PASSWORD  "val"
	#define DATABASE  "edc"
	/////////////////////////////////////////////////////////////////////////////
	// CAboutDlg dialog used for App About

	class CAboutDlg : public CDialog
	{
	public:
		CAboutDlg();

	// Dialog Data
		//{{AFX_DATA(CAboutDlg)
		enum { IDD = IDD_ABOUTBOX };
		//}}AFX_DATA

		// ClassWizard generated virtual function overrides
		//{{AFX_VIRTUAL(CAboutDlg)
		protected:
		virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
		//}}AFX_VIRTUAL

	// Implementation
```

```
protected:
	//{{AFX_MSG(CAboutDlg)
	virtual void OnOK();
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
	//{{AFX_DATA_INIT(CAboutDlg)
	//}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CAboutDlg)
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
	//{{AFX_MSG_MAP(CAboutDlg)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////

// CMDNDlg dialog

CMDNDlg::CMDNDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CMDNDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CMDNDlg)
	m_EdtPekerjaan = _T("");
	m_EdtIdCrd = _T("");
	m_EdtSaldoAwal = _T("");
	m_EdtAlamat = _T("");
	m_EdtTglLahir = _T("");
	m_EdtTempatLahir = _T("");
	m_EdtTlp = _T("");
	m_EdtNoRek = _T("");
	m_EdtNama = _T("");
	m_EdtIdNasabah = _T("");
	//}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CMDNDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CMDNDlg)
	DDX_Control(pDX, IDC_CMB_STATUS, m_CmbStatus);
	DDX_Text(pDX, IDC_EDT_PKRJN, m_EdtPekerjaan);
	DDX_Text(pDX, IDC_EDT_NOREK, m_EdtNoRek);
	DDX_Text(pDX, IDC_EDT_NM, m_EdtNama);
	DDX_Text(pDX, IDC_EDT_IDNAS, m_EdtIdNasabah);
	DDX_Text(pDX, IDC_EDT_IDCRD, m_EdtIdCrd);
	DDX_Text(pDX, IDC_EDT_AWL, m_EdtSaldoAwal);
	DDX_Text(pDX, IDC_EDT_ALMT, m_EdtAlamat);
	DDX_Text(pDX, IDC_EDT_TGLLHR, m_EdtTglLahir);
	DDX_Text(pDX, IDC_EDT_TLHR, m_EdtTempatLahir);
	DDX_Text(pDX, IDC_EDT_TLP, m_EdtTlp);
	//}}AFX_DATA_MAP
```

```
}

BEGIN_MESSAGE_MAP(CMDNDlg, CDialog)
	//{{AFX_MSG_MAP(CMDNDlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_BTN_SIMPAN, OnBtnSimpan)
	ON_BN_CLICKED(IDC_BTN_CARI, OnBtnCari)
	ON_BN_CLICKED(IDC_BTN_NEXT, OnBtnNext)
	ON_BN_CLICKED(IDC_BTN_PERBAIKI, OnBtnPerbaiki)
	ON_BN_CLICKED(IDC_BTN_HPS, OnBtnHps)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMDNDlg message handlers

BOOL CMDNDlg::OnInitDialog()
{
	CDialog::OnInitDialog();

	// Add "About..." menu item to system menu.

	// IDM_ABOUTBOX must be in the system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX,
strAboutMenu);
		}
	}

	// Set the icon for this dialog.  The framework does this automatically
	//  when the application's main window is not a dialog
	SetIcon(m_hIcon, TRUE);			// Set big icon
	SetIcon(m_hIcon, FALSE);		// Set small icon

	// TODO: Add extra initialization here
	//MYSQL *conn;
	conn = mysql_init(NULL);
	mysql_real_connect(conn, LOCALHOST, USERNAME, PASSWORD, DATABASE, 0,
NULL, 0);

	return TRUE;  // return TRUE  unless you set the focus to a control
}

void CMDNDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)
	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	}
	else
	{
```

```
                CDialog::OnSysCommand(nID, lParam);
        }
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CMDNDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for painting

                SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CMDNDlg::OnQueryDragIcon()
{
        return (HCURSOR) m_hIcon;
}

void CMDNDlg::OnBtnSimpan()
{
        // TODO: Add your control notification handler code here
        UpdateData(TRUE);

        CString sStatus;

        mysql_query(conn, "INSERT INTO Nasabah (Id_Nasabah, Id_Kartu, Nama, Alamat,
Tempat_Lahir, Tanggal_Lahir, Telephone, Pekerjaan, Jenis_Kelamin, Status) values
(m_EdtIdNasabah, m_EdtIdCrd, m_EdtNama, m_EdtAlamat, m_EdtTempatLahir, m_EdtTglLahir,
m_EdtTlp, m_EdtPekerjaan, \"L\", sStatus)");
}

void CMDNDlg::OnBtnCari()
{
        // TODO: Add your control notification handler code here
        UpdateData(TRUE);
        CString sStatus;

        mysql_query(conn, "SELECT * FROM Nasabah WHERE Id_Nasabah = m_EdtIdCrd");
        /
}

void CMDNDlg::OnBtnNext()
```

```
{
	// TODO: Add your control notification handler code here
	UpdateData(TRUE);
	CString sStatus;

	mysql_query(conn, "SELECT COUNT(*) FROM Nasabah WHERE Id_Kartu =
m_EdtIdCrd");
}

void CMDNDlg::OnBtnPerbaiki()
{
	// TODO: Add your control notification handler code here
	UpdateData(TRUE);
	CString sStatus;

	mysql_query(conn, "REPAIR FROM Nasabah WHERE Id_Kartu = m_EdtIdCrd");
}

void CMDNDlg::OnBtnHps()
{
	// TODO: Add your control notification handler code here
	UpdateData(TRUE);
	CString sStatus;

	mysql_query(conn, "DELETE FROM Nasabah WHERE Id_Nasabah = m_EdtIdCrd");
}

void CAboutDlg::OnOK()
{
	// TODO: Add extra validation here

	CDialog::OnOK();
}
```

- **EDC (*Electronic Data Capture*)**

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

/*************************************************
This program was produced by the
CodeWizardAVR V1.25.3 Standard
Automatic Program Generator
© Copyright 1998-2007 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 7/21/2009
Author  : F4CG
Company : F4CG
Comments:


Chip type           : ATmega162
Program type        : Application
Clock frequency     : 11.059000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*************************************************/
```

```
#include <mega162.h>

// I2C Bus functions
#asm
   .equ __i2c_port=0x18 ;PORTB
   .equ __sda_bit=0
   .equ __scl_bit=1
#endasm
#include <i2c.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x1B ;PORTA
#endasm
#include <lcd.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART0 Receiver buffer
#define RX_BUFFER_SIZE0 8
char rx_buffer0[RX_BUFFER_SIZE0];

#if RX_BUFFER_SIZE0<256
unsigned char rx_wr_index0,rx_rd_index0,rx_counter0;
#else
unsigned int rx_wr_index0,rx_rd_index0,rx_counter0;
#endif

// This flag is set on USART0 Receiver buffer overflow
bit rx_buffer_overflow0;

// USART0 Receiver interrupt service routine
interrupt [USART0_RXC] void usart0_rx_isr(void)
{
char status,data;
status=UCSR0A;
data=UDR0;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer0[rx_wr_index0]=data;
   if (++rx_wr_index0 == RX_BUFFER_SIZE0) rx_wr_index0=0;
   if (++rx_counter0 == RX_BUFFER_SIZE0)
      {
      rx_counter0=0;
      rx_buffer_overflow0=1;
      };
   };
            //rcv data from POS
            i=0;
            while(rx_counter0==1)
            {
```

```
            datapos[i]=getchar0();
            i=i+1;
            }

}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART0 Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter0==0);
data=rx_buffer0[rx_rd_index0];
if (++rx_rd_index0 == RX_BUFFER_SIZE0) rx_rd_index0=0;
#asm("cli")
--rx_counter0;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART0 Transmitter buffer
#define TX_BUFFER_SIZE0 8
char tx_buffer0[TX_BUFFER_SIZE0];

#if TX_BUFFER_SIZE0<256
unsigned char tx_wr_index0,tx_rd_index0,tx_counter0;
#else
unsigned int tx_wr_index0,tx_rd_index0,tx_counter0;
#endif

// USART0 Transmitter interrupt service routine
interrupt [USART0_TXC] void usart0_tx_isr(void)
{
if (tx_counter0)
   {
   --tx_counter0;
   UDR0=tx_buffer0[tx_rd_index0];
   if (++tx_rd_index0 == TX_BUFFER_SIZE0) tx_rd_index0=0;
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART0 Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter0 == TX_BUFFER_SIZE0);
#asm("cli")
if (tx_counter0 || ((UCSR0A & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer0[tx_wr_index0]=c;
   if (++tx_wr_index0 == TX_BUFFER_SIZE0) tx_wr_index0=0;
   ++tx_counter0;
   }
else
   UDR0=c;
#asm("sei")
}
#pragma used-
```

```
#endif

// USART1 Receiver buffer
#define RX_BUFFER_SIZE1 8
char rx_buffer1[RX_BUFFER_SIZE1];

#if RX_BUFFER_SIZE1<256
unsigned char rx_wr_index1,rx_rd_index1,rx_counter1;
#else
unsigned int rx_wr_index1,rx_rd_index1,rx_counter1;
#endif

// This flag is set on USART1 Receiver buffer overflow
bit rx_buffer_overflow1;

// USART1 Receiver interrupt service routine
interrupt [USART1_RXC] void usart1_rx_isr(void)
{
char status,data;
status=UCSR1A;
data=UDR1;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer1[rx_wr_index1]=data;
   if (++rx_wr_index1 == RX_BUFFER_SIZE1) rx_wr_index1=0;
   if (++rx_counter1 == RX_BUFFER_SIZE1)
      {
      rx_counter1=0;
      rx_buffer_overflow1=1;
      };
   };

  char data;


}

// Get a character from the USART1 Receiver buffer
#pragma used+
char getchar1(void)
{
char data;
while (rx_counter1==0);
data=rx_buffer1[rx_rd_index1];
if (++rx_rd_index1 == RX_BUFFER_SIZE1) rx_rd_index1=0;
#asm("cli")
--rx_counter1;
#asm("sei")
return data;
}
#pragma used-
// USART1 Transmitter buffer
#define TX_BUFFER_SIZE1 8
char tx_buffer1[TX_BUFFER_SIZE1];

#if TX_BUFFER_SIZE1<256
unsigned char tx_wr_index1,tx_rd_index1,tx_counter1;
#else
unsigned int tx_wr_index1,tx_rd_index1,tx_counter1;
#endif

// USART1 Transmitter interrupt service routine
interrupt [USART1_TXC] void usart1_tx_isr(void)
{
```

```
if (tx_counter1)
   {
   --tx_counter1;
   UDR1=tx_buffer1[tx_rd_index1];
   if (++tx_rd_index1 == TX_BUFFER_SIZE1) tx_rd_index1=0;
   };
}

// Write a character to the USART1 Transmitter buffer
#pragma used+
void putchar1(char c)
{
while (tx_counter1 == TX_BUFFER_SIZE1);
#asm("cli")
if (tx_counter1 || ((UCSR1A & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer1[tx_wr_index1]=c;
   if (++tx_wr_index1 == TX_BUFFER_SIZE1) tx_wr_index1=0;
   ++tx_counter1;
   }
else
   UDR1=c;
#asm("sei")
}
#pragma used-

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here
int j;
unsigned int sid;
unsigned int reg;
unsigned int pEEPROMData; char a;
char d[8];
char e[20];
char f[20];
char ok;
char rHandle[]={"ACR120_open(ACR120_COM, ACR120_COM_BAUDRATE_9600)"};
unsigned int
pEEPROMData[18]={0x00,0x01,0x02,0x03,0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x1
9,0x1A,0x1B,0x1C,0x1D};
void main(void)
{
// Declare your local variables here

// Crystal Oscillator division factor: 1
#pragma optsize-
CLKPR=0x80;
CLKPR=0x00;
#ifdef _OPTIMIZE_SIZE_
#pragma optsize+
#endif

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
```

```
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0xff;
DDRC=0x0f;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;

// Port E initialization
// Func2=In Func1=In Func0=In
// State2=T State1=T State0=T
PORTE=0x00;
DDRE=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// Timer/Counter 3 initialization
```

```
// Clock value: Timer 3 Stopped
// Mode: Normal top=FFFFh
// Noise Canceler: Off
// Input Capture on Falling Edge
// OC3A output: Discon.
// OC3B output: Discon.
// Timer 3 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR3A=0x00;
TCCR3B=0x00;
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x00;
OCR3AL=0x00;
OCR3BH=0x00;
OCR3BL=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
// Interrupt on any change on pins PCINT0-7: Off
// Interrupt on any change on pins PCINT8-15: Off
MCUCR=0x00;
EMCUCR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;
ETIMSK=0x00;

// USART0 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
// USART0 Transmitter: On
// USART0 Mode: Asynchronous
// USART0 Baud rate: 9600
UCSR0A=0x00;
UCSR0B=0xD8;
UCSR0C=0x86;
UBRR0H=0x00;
UBRR0L=0x47;

// USART1 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART1 Receiver: On
// USART1 Transmitter: On
// USART1 Mode: Asynchronous
// USART1 Baud rate: 9600
UCSR1A=0x00;
UCSR1B=0xD8;
UCSR1C=0x86;
UBRR1H=0x00;
UBRR1L=0x47;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;

// I2C Bus initialization
```

```
i2c_init();

// LCD module initialization
lcd_init(16);

// GPRS Init
initGPRS();

// Global enable interrupts
#asm("sei")

while (1)
      {
      // Place your code here
      ulang:
      lcd_gotoxy(0,0);
      lcd_putsf("Dekatkan Kartu");
      lcd_gotoxy(0,1);
      lcd_putsf("Ke Reader");
      delay_ms(500);
      bacaFrameRcvSMCrd();
      lcd_gotoxy(0,0);
      lcd_putsf("Masukkan PIN");
      delay_ms(500);
      for (j=0;j<=3;j++)
      {
      ulang1;
      if (PINC.4=1 && PINC.5=1 && PINC.6=1 && PINC.7=1){goto ulang1};
      inputkeypad();
      f[j]=a;
      lcd_gotoxy(j,0);
      lcd_putchar(a);
      delay_ms(1200);
      }
      initGPRS();
      kar();  //get id
      sendGSM(pCmd);
      initreceived();
      getGSM();
      };
}

char gTmpBuf[16];       // global temporary buffer
char gGTxBuffer[16];

void initGPRS()
{
   // Place your code here
   sendGSM("ATE=0\r");
   sendGSM("AT+CGDCONT=1,\"192.1.0.10 \",\"http://www.xlwap.co.id\"");
   sendGSM("+ATDT#99*")
}
void kar()
{
char *pCmd;
for(i=4;i<=17;i++)
{
pCmd[i]=d[i];
}
return pCmd;
 }

void sendGSM(char *pCmd)
{
```

```
        int finish;
        finish = 1;
        strcpy(gTmpBuf, pCmd);
        putsGSM();
        strcpy(gTmpBuf, datapos);
        putsGSM();
        clearDataPos();
        do
        {
          // get error code from serial

        }while(finish != 1);
}

void clearDataPos();
{
datapos=" ";
}

void putsGSM(void)
{
        int i;
        int len = strlen(gTmpBuf);
        for(i=0; i<len; i++)
        {
          putchar(gTmpBuf[i]);
        }
}




void initreceived()
{
        sendGSM("ATCMGF=0\r");
        sendGSM("AT+CNMICONT=1,\"http://xlwap.co.id\",\"internet\"");
        sendGSM("+ATDT#99*");
}

void getGSM(void)
{
ok=getchar();
if (ok=='1'){lcd_gotoxy(0,0);lcd_putsf("Transaksi");lcd_gotoxy(0,1);lcd_putsf("Selesai")} else
{lcd_gotoxy(0,0);lcd_putsf("Saldo");lcd_gotoxy(0,1);lscd_putsf("Kurang");}
}

void bacaFrameRcvSMCrd()
{
        sid=1;
        reg=6;
        ulang:
        for (j=0;j<10;j++)
        {
        printf("ACR120_Select(%s,%d,0x03,0x03,0x03)",rHandle,sid);
        e[j]=getchar();
        }
        if (e[2]=='c' e[3]=='6' e[4]=='2' e[5]=='B' e[6]=='2' e[7]=='A' e[8]=='9' e[9]=='9')
        bacaFrameRcvSMCrd1(); else goto ulang;
        }

  void bacaFrameRcvSMCrd1()
        lanjut:
        for(j=0;j<18;j++)
        {
```

```
      printf("ACR120_ReadEEPROM (%s, %d, %d, %d)",rHandle,reg,sid,pEEPROMData[i]);
      d[j]=getchar();
      }
      if (d[0]==f[0] && d[1]==f[1] && d[2]==f[2] && d[3]==f[3])
      {
      if (d[4]='1'  d[5]=='2' && d[6]=='3' && d[7]=='4' && d[8]=='5' d[9]=='6' && d[10]=='7' &&
d[11]=='8' && d[12]=='9'
        d[13]='1'  d[14]=='0' && d[15]=='0' && d[16]=='0' && d[17]=='1') putsGSM(); else goto
ulang;}
};

void inputkeypad()
{
PORTC=0b11111110;
if (PINC.4==0 && PINC.5==1 && PINC.6==1 && PINC.7==1 ) {lcd_putsf("*");a='1';}
else if (PINC.4==1 && PINC.5==0 && PINC.6==1 && PINC.7==1 ) {lcd_putsf("*");a='4';}
else if (PINC.4==1 && PINC.5==1 && PINC.6==0 && PINC.7==1 ) {lcd_putsf("*");a='7';}else
PORTC=0b11111101;
if (PINC.4==0 && PINC.5==1 && PINC.6==1 && PINC.7==1 ) {lcd_putsf("*");a='2';}
else if (PINC.4==1 && PINC.5==0 && PINC.6==1 && PINC.7==1 ) {lcd_putsf("*");a='5';}
else if (PINC.4==1 && PINC.5==1 && PINC.6==0 && PINC.7==1 ) {lcd_putsf("*");a='8';}
else if (PINC.4==1 && PINC.5==1 && PINC.6==1 && PINC.7==0 ) {lcd_putsf("*");a='0';} else
PORTC=0b11111011;
if (PINC.4==0 && PINC.5==1 && PINC.6==1 && PINC.7==1 ) {lcd_putsf("*");a='3';}
else if (PINC.4==1 && PINC.5==0 && PINC.6==1 && PINC.7==1 ) {lcd_putsf("*");a='6';}
else if (PINC.4==1 && PINC.5==1 && PINC.6==0 && PINC.7==1 ) {lcd_putsf("*");a='9';}

}
```