Додаток А

Текст програми CorrelationCode

```
#include "stdafx.h"
#include "C2.h"
#include "C2Dlg.h"

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

/////////////////////////////////////////////////////////////////////////////
// CC2App

BEGIN_MESSAGE_MAP(CC2App, CWinApp)
      //{{AFX_MSG_MAP(CC2App)
      // 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()

/////////////////////////////////////////////////////////////////////////////
// CC2App construction

CC2App::CC2App()
{
      // TODO: add construction code here,
      // Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CC2App object

CC2App theApp;

/////////////////////////////////////////////////////////////////////////////
// CC2App initialization

BOOL CC2App::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

      CC2Dlg 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;
}


// C2Dlg.cpp : implementation file
//

#include "stdafx.h"
#include "C2.h"
#include "C2Dlg.h"

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

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

#define len 27
#define min_p -1

int s=0, min_pelustka=min_p;

char mas[len];
char res[4];

void rotate(char *mas, int bit);
void itoc (int a, char *mas);
void print(char *xPtr, int bit);

int korel(char *mas, const int bit)
{
char etalon[len], *temp=mas;
int k_res[len], result=0;
for(int i=0;i<bit;i++) etalon[i]=*mas, mas++; // copy string
mas=temp;
	for(i=0;i<len;i++) k_res[i]=0; // clear massive
	for(int x=0;x<bit; x++)
	{
	for(i=0;i<bit;i++)
	{
		if(*mas==etalon[i])	k_res[x]++;
		mas++;
	}
	mas=temp;
	rotate(mas, bit);
	}
```

```
//for(i=0;i<bit;i++)    itoc(k_res[i], &res[0]), print(&res[0], 3);

for(i=1;i<bit;i++)
{
if(k_res[i]>result) result=k_res[i];
if(k_res[i]<min_pelustka) min_pelustka=k_res[i];
}
result-=(len-result);
return result;
}

int compare(char *mas1, char *mas2, int bit)
{
      int result=0;
      for(;bit!=0;bit--)
      {
            if(*mas1==*mas2) result++;
            mas1++, mas2++;
      }
return result;
}

void rotate(char *mas, int bit)
{
      char symbol, temp;
      symbol=*mas;
      for(int i=0; i<bit; i++) mas++, temp=*mas, mas--, *mas=temp, mas++;
      mas--;
      *mas=symbol;
}

void ltob(char *string, long value)
{
int counter=len;
string+=len-1;
for(; counter!=-1; counter--)
      {
      if(value&1) *string='1';
      else *string='0';
      string--;
      value>>=1;
      }
}
/*
// function printed string
void print(char *xPtr, int bit)
{
      for(int i=0; i<bit; i++) cout<<*xPtr, xPtr++;
      cout<<endl;
}
*/
void itoc (int a, char *mas)
{
char *temp=mas+1, z;
if(a>=0) *mas=' ';
else *mas='-', a*=-1;
      mas++;
      for (int x=0; x<3; x++) *mas='0', mas++;
      z=a/100, a=a-(z*100), *temp+=z, temp++;
      z=a/10, a=a-(z*10), *temp+=z, temp++;
      z=a, *temp+=z;
}
```

```
struct korel
{
//public:
char text[32];
char tb1;
char pik [4];
char tb2;
char max_pelustka[4];
char tb3;
char bit[2];
char tb4;
char znach[10];
char endl;
};



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()

/////////////////////////////////////////////////////////////////////////////
```

```
// CC2Dlg dialog

CC2Dlg::CC2Dlg(CWnd* pParent /*=NULL*/)
	: CDialog(CC2Dlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CC2Dlg)
	m_Two = FALSE;
	m_V = 0;
	m_Max = 0;
	m_Min = 0;
	m_H = 0;
	//}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CC2Dlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CC2Dlg)
	DDX_Control(pDX, IDC_STATIC_Method, m_Method);
	DDX_Control(pDX, IDC_PROGRESS1, m_Progress);
	DDX_Check(pDX, IDC_CHECK_TWO_DIM, m_Two);
	DDX_Text(pDX, IDC_EDIT_V, m_V);
	DDX_Text(pDX, IDC_EDIT_Max, m_Max);
	DDX_Text(pDX, IDC_EDIT_Min, m_Min);
	DDX_Text(pDX, IDC_EDIT_H, m_H);
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CC2Dlg, CDialog)
	//{{AFX_MSG_MAP(CC2Dlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_CHECK_TWO_DIM, OnCheckTwoDim)
	ON_BN_CLICKED(IDC_BUTTON_Exit, OnBUTTONExit)
	ON_BN_CLICKED(IDC_BUTTON_File, OnBUTTONFile)
	ON_BN_CLICKED(IDC_BUTTON_Start, OnBUTTONStart)
	ON_BN_CLICKED(IDC_BUTTON_Stop, OnBUTTONStop)
	ON_BN_CLICKED(IDC_BUTTON_MAX_PM, OnButtonMaxPm)
	ON_BN_CLICKED(IDC_BUTTON_MAX_PP, OnButtonMaxPp)
	ON_BN_CLICKED(IDC_BUTTON_HM, OnButtonHm)
	ON_BN_CLICKED(IDC_BUTTON_HP, OnButtonHp)
	ON_BN_CLICKED(IDC_BUTTON_MIN_PM, OnButtonMinPm)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CC2Dlg message handlers

BOOL CC2Dlg::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 CC2Dlg::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 CC2Dlg::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
```

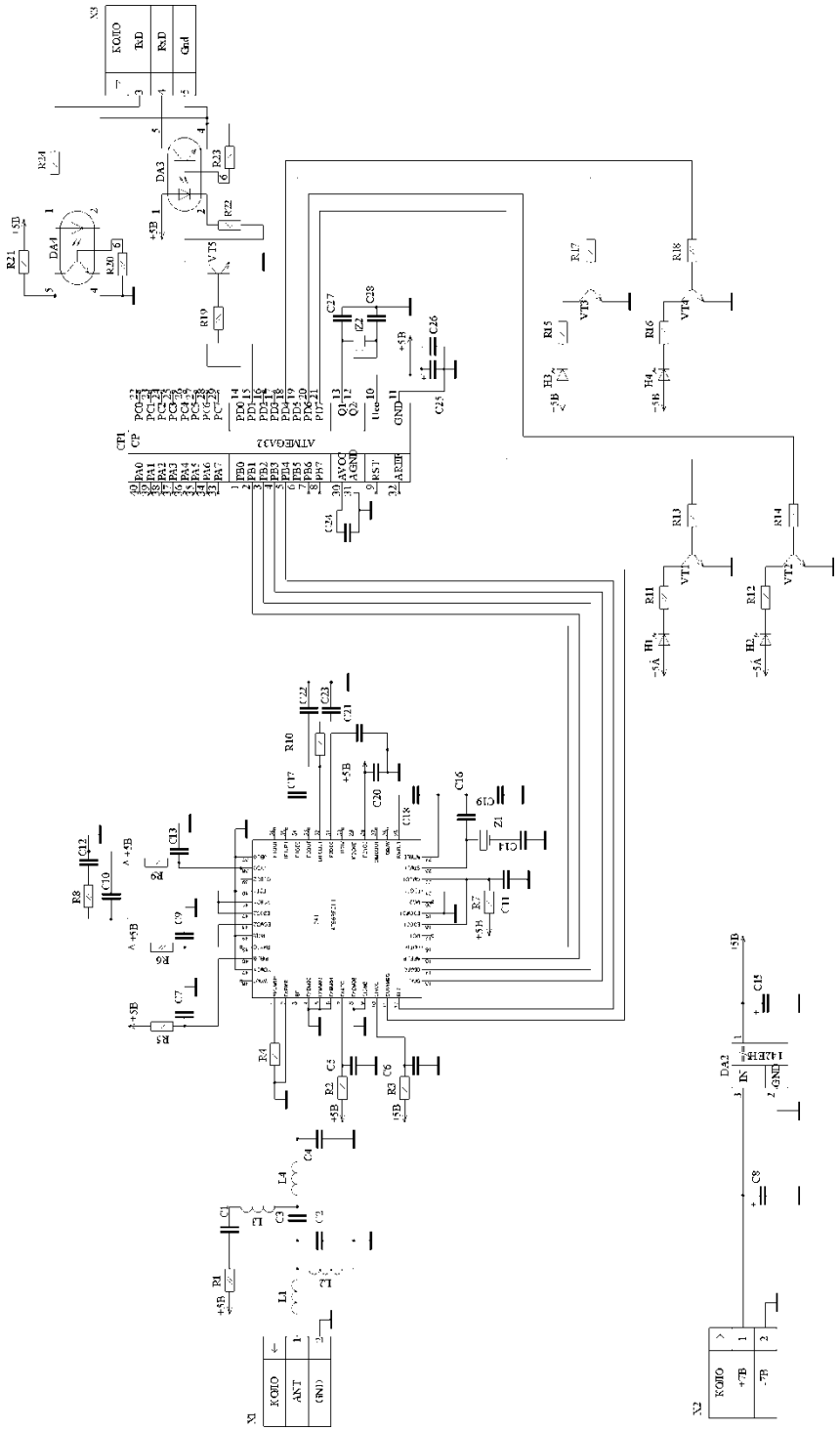
```
//  the minimized window.
HCURSOR CC2Dlg::OnQueryDragIcon()
{
      return (HCURSOR) m_hIcon;
}

void CC2Dlg::OnCheckTwoDim()
{
      // TODO: Add your control notification handler code here
      UpdateData(TRUE);
      if(m_Two == TRUE)
      GetDlgItem (IDC_EDIT_V)->EnableWindow(SW_SHOW);
      else
      GetDlgItem (IDC_EDIT_V)->EnableWindow(SW_HIDE);
      //UpdateData(FALSE);

}

void CC2Dlg::OnBUTTONExit()
{
      // TODO: Add your control notification handler code here

      OnOK();

}

void CC2Dlg::OnBUTTONFile()
{
      // TODO: Add your control notification handler code here
}

void CC2Dlg::OnBUTTONStart()
{

//    char m_txt[44];
      struct korel x;
//    int len;
      UpdateData(TRUE);
//    len = m_H;

      CFileDialog       DlgSaveAs       (FALSE,       (LPCSTR)"txt",       NULL,
      OFN_HIDEREADONLY|OFN_OVERWRITEPROMPT,
      (LPCSTR)"Text Files (*.txt) |*.txt||");
      if(DlgSaveAs.DoModal()==IDOK){
      CStdioFile File (DlgSaveAs.GetPathName(),
            CFile::modeCreate|CFile::modeWrite|CFile::typeBinary);

//-----------------------------------------------------------------

x.tb1=9;
x.tb2=9;
x.endl=13;
x.tb3=9;
x.tb4=9;
x.bit[0]='0';
x.bit[1]='0';

/*
ltob(&x.text[32-len], 5);
print(&x.text[32-len],len);
*/
//x.data='5';
int z=1, bit=len, zx;
```

```
for(int i=0; i<len; i++) z*=2;

m_Progress.SetRange32( 0, z);

for(i=0; i<z; i++)
      {
for(int i3=0; i3<32; i3++) x.text[i3]=' ';
for(i3=0; i3<10; i3++) x.znach[i3]=' ';
      zx=s;
      itoa(zx, x.znach, 10);
      ltob(&x.text[32-len], s);
//    print(&x.text[32-len], len);
//    itoc(korel(&x.text[32-len], len), &x.max_pelustka[0]);
      x.max_pelustka[0]=korel(&x.text[32-len], len);
      itoc(len, &x.pik[0]);
//    print(&x.text[32-len], len);
      s++;
      for(int i1=0; x.text[32-len+i1]!='1'; i1++)    /*if(x.text[32-
      len+i1]!='1') */bit--;

      itoa(bit, x.bit, 10);
bit = len;
x.tb4 = 9;
if(x.max_pelustka[0] == -1 && min_pelustka == -1)
{
//    File.WriteString ((LPCTSTR)m_txt);//fwrite(&x, sizeof(x), 1, stream); //
      write struct s to file
      itoa(x.max_pelustka[0], &x.max_pelustka[0], 10);
      File.Write (&x, sizeof (x));
}
// ------- Progress Bar -----------------

m_Progress.SetPos(i);

// ------End Progress Bar -----------------


//    MessageBox(x.znach);
      }



//-----------------------------------------------------------------
/*
      File.WriteString ((LPCTSTR)m_txt);
      File.WriteString ((LPCTSTR)m_txt);
      File.WriteString ((LPCTSTR)m_txt); */
      }

}

void CC2Dlg::OnBUTTONStop()
{

// TODO: Add your control notification handler code here
//    UpdateData(TRUE);

}

void CC2Dlg::OnButtonMaxPm()
{
      m_Max--;
      UpdateData(FALSE);
```

```
}

void CC2Dlg::OnButtonMaxPp()
{
	m_Max++;
	UpdateData(FALSE);
}

void CC2Dlg::OnButtonHm()
{
	m_H--;
	UpdateData(FALSE);
}


void CC2Dlg::OnButtonHp()
{
	m_H++;
	UpdateData(FALSE);
}

void CC2Dlg::OnButtonMinPm()
{
	m_Min--;
	UpdateData(FALSE);
}
```

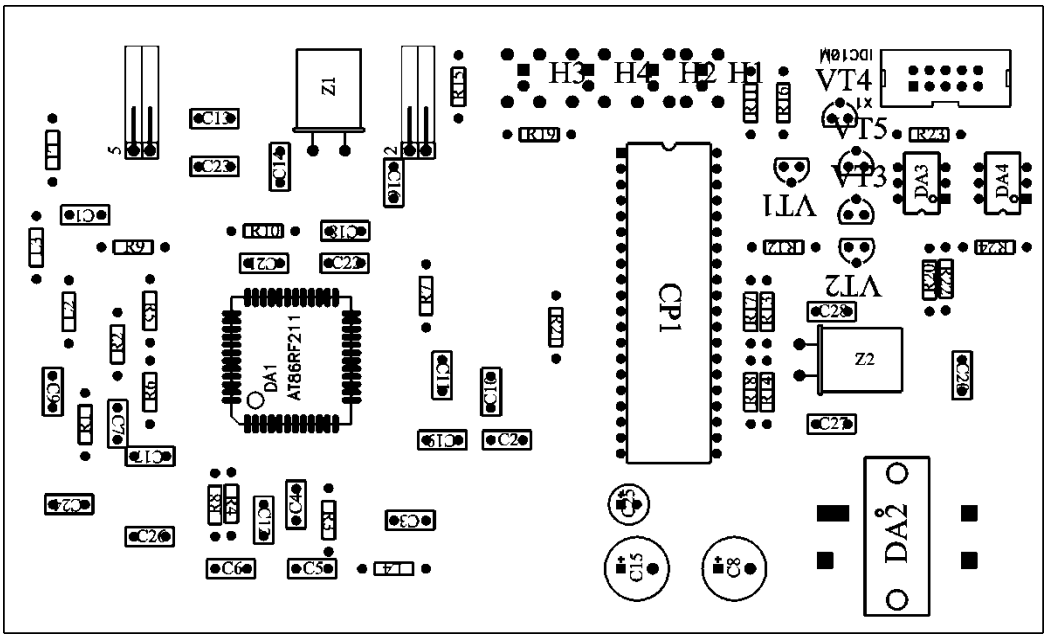
Додаток Б

Прийомопередавач. Схема електрична принципова



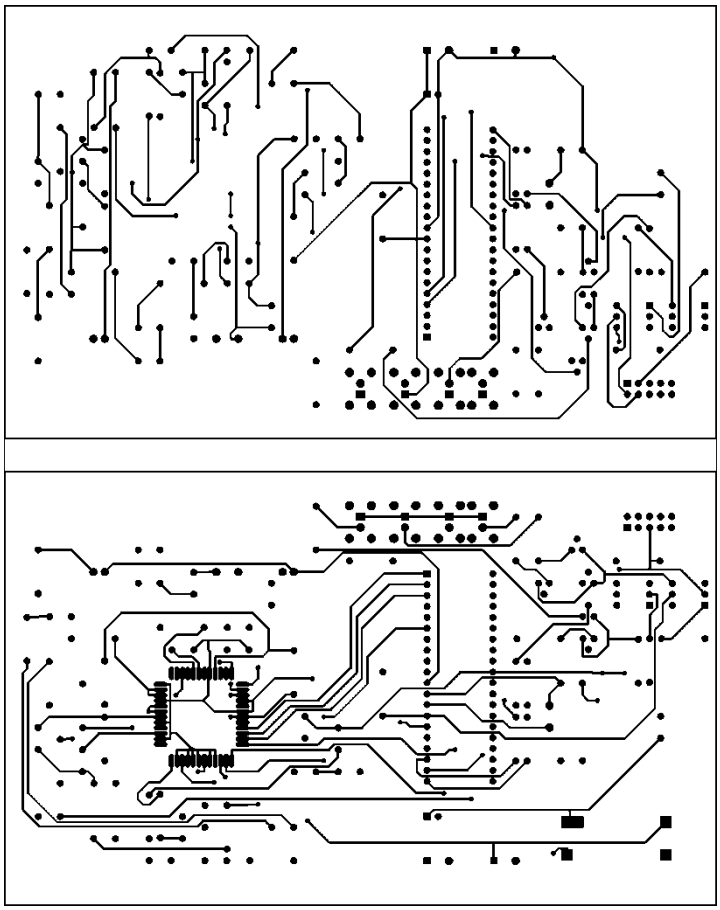
Додаток В

Прийомопередавач. Розміщення елементів



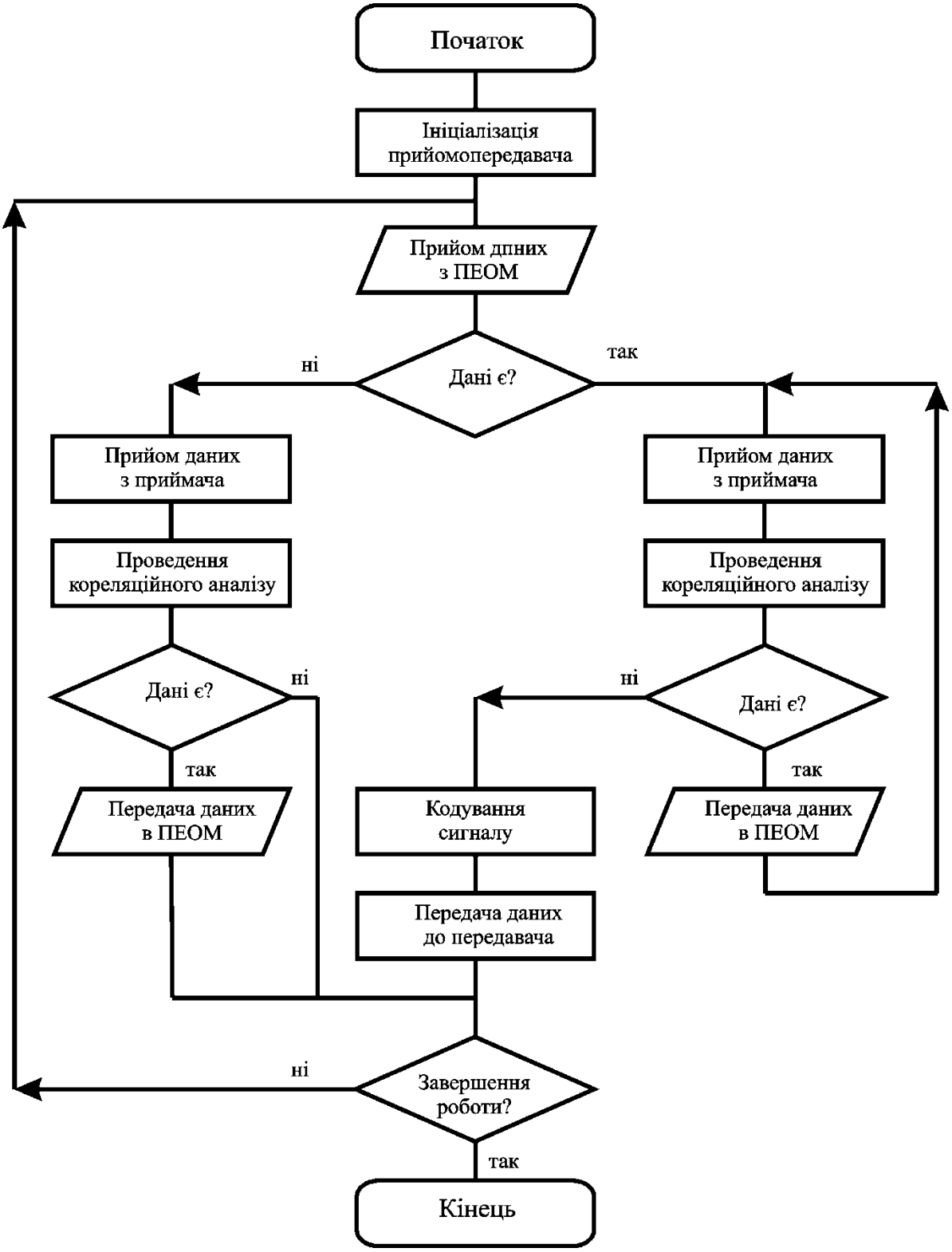
Додаток Г

Прийомопередавач. Плата друкована



Додаток Д

Блок схема роботи програми мікроконтролера



Додаток Е

Акт впровадження

Додаток К

Копія публікації