



```
//-----unit1.cpp
```

```
#include <vcl.h>
#include <stdio.h>
#include <stdlib.h>
```

```
#pragma hdrstop
```

```
#include "Unit1.h"
#include "Graph.h"
#include "PCI.h"
```

```
//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
```

```
//-----
__fastcall TForm1::TForm1(TComponent* Owner)
: TForm(Owner)
{
}
```

```
//-----
void __fastcall TForm1::FormCreate(TObject *Sender)
{
    pci_open(); //全ボード OPEN
```

```
pci3523DA_set(1);
da_out(0.0,1);
pci3523DA_set(2);
da_out(0.0,2);
pci3523DA_set(3);
da_out(0.0,3);
pci3523DA_set(4);
da_out(0.0,4);
```

```
}
```

```
//-----
```

```
//サンプリング時間のボタン
```

```
void __fastcall TForm1::Button1Click(TObject *Sender)
```

```
{
/* timer(変換ボタン) */
  ts=StrToFloat(Edit1->Text);
  if(ts>0.5){
    ts=0.5;
  }
}
```

```
RadioGroup1->Enabled=true;
```

```
}
```

```
//-----
```

```
void __fastcall TForm1::RadioGroup1Click(TObject *Sender)
```

```
{
GroupBox2->Enabled=true;
GroupBox3->Enabled=true;
```

```
    if(RadioGroup1->ItemIndex==0){          /*AD*/
```

```
        Button2->Enabled=true;
        GroupBox4->Enabled=true;
        RadioGroup2->Enabled=true;
    }
```

```
    else if(RadioGroup1->ItemIndex==1){     /*DA*/
```

```
        Button3->Enabled=true;
        GroupBox5->Enabled=true;
        RadioGroup3->Enabled=true;
    }
```

```
    else if(RadioGroup1->ItemIndex==2){     /*PCC*/
```

```
        Button4->Enabled=true;
        GroupBox6->Enabled=true;
        RadioGroup4->Enabled=true;
    }
```

```
    else if(RadioGroup1->ItemIndex==3){     /*AD-DA*/
```

```
        Button5->Enabled=true;
        GroupBox7->Enabled=true;
        RadioGroup2->Enabled=true;
        RadioGroup3->Enabled=true;
    }
```

```
}
```

```
//-----
```

```
//AD チェックのボタン
```

```
void __fastcall TForm1::Button2Click(TObject *Sender)
```

```
{
  static int i,i1,chn;
  int button;
```

```
Button6->Enabled=true;
```

```
    if(RadioGroup2->ItemIndex==0)
```

```
        chn=1;
```

```
    else if(RadioGroup2->ItemIndex==1)
```

```
        chn=2;
```

```
    else if(RadioGroup2->ItemIndex==2)
```

```
        chn=3;
```

```
    else if(RadioGroup2->ItemIndex==3)
```

```
        chn=4;
```

```

pci3523AD_set(chn);
count=StrToInt(Edit2->Text);

for(i=0;i<=count;i++){

    tim_sync();                //タイマ同期
    ad_value[i]=ad_in(chn);    //A/D 入力

}
button = Application->MessageBox("Finish!! Graph is OK and Memo is cancel","Status", MB_OKCANCEL);
if (button == IDOK)
{
    Graph->ShowModal();
}
if (button == IDCANCEL)
{
    Memo1->Lines->Add("AD DATA");
    Memo1->Lines->Add(" ");
    for(i1=1;i1<=count;i1++)
        Memo1->Lines->Add(FloatToStr(ad_value[i1]));
}
}

//-----
//DA チェックのボタン
void __fastcall TForm1::Button3Click(TObject *Sender)
{
int DACH,button;
float vo;

if(RadioGroup3->ItemIndex==0)
    DACH=1;
else if(RadioGroup3->ItemIndex==1)
    DACH=2;
else if(RadioGroup3->ItemIndex==2)
    DACH=3;
else if(RadioGroup3->ItemIndex==3)
    DACH=4;

pci3523DA_set(DACH);
vo=StrToFloat(Edit3->Text);

da_out(vo,DACH);                //D/A 出力

button = Application->MessageBox("D/A complete!! if OK then D/A clear","Status", MB_OK);

    if (button == IDOK) {
        da_out(0.0,DACH);
    }
}

//-----
//PCC120 チェックのボタン
void __fastcall TForm1::Button4Click(TObject *Sender)
{
static int button,i1,pccCH;
static long enc1;

Button6->Enabled=true;

if(RadioGroup4->ItemIndex==0)
    pccCH=1;
else if(RadioGroup4->ItemIndex==1)
    pccCH=2;
else if(RadioGroup4->ItemIndex==2)
    pccCH=3;
else if(RadioGroup4->ItemIndex==3)
    pccCH=4;
if ( pccCH>=3 ) {
    pccCH==2;
}
}

```

```

Form1->Memo1->Lines->Add("CH1 and CH2 are setable");
Form1->Memo1->Lines->Add("Now CH2 are set");
}
count=StrToInt(Edit4->Text);

pci6204_set(pccCH); // 2相パルス、4 通倍 など設定

for(i1=0;i1<=count;i1++){
    tim_sync(); // タイマ同期
    enc1=pci6204_get(pccCH); // カウントパルス読み込み
    pcc_value[i1]=(float)enc1;
}

button = Application->MessageBox("Finish!! Graph is OK and Memo is cancel","Status", MB_OKCANCEL);
if (button == IDOK)
{
    Graph->ShowModal();
}
if (button == IDCANCEL)
{
    Memo1->Lines->Add("PCC DATA");
    Memo1->Lines->Add(" ");
    for (i1=1;i1<=count;i1++)
        Memo1->Lines->Add(FloatToStr(pcc_value[i1]));
}
}
//-----

void __fastcall TForm1::Button5Click(TObject *Sender)
{
    static double volt1,volt2,volt3,volt4,vo,amp,offset,prd;
    static int i,j,DACH,kind,ci,cih,button,timer,pccCH,chn,mod=0;
    static long enc1;

    Button6->Enabled=true;

    if(RadioGroup2->ItemIndex==0)
        chn=1;
    else if(RadioGroup2->ItemIndex==1)
        chn=2;
    else if(RadioGroup2->ItemIndex==2)
        chn=3;
    else if(RadioGroup2->ItemIndex==3)
        chn=4;

    if(RadioGroup3->ItemIndex==0)
        DACH=1;
    else if(RadioGroup3->ItemIndex==1)
        DACH=2;
    else if(RadioGroup3->ItemIndex==2)
        DACH=3;
    else if(RadioGroup3->ItemIndex==3)
        DACH=4;

    /* if(RadioGroup4->ItemIndex==0)
        pccCH=1;
    else if(RadioGroup4->ItemIndex==1)
        pccCH=2;
    else if(RadioGroup4->ItemIndex==2)
        pccCH=3;
    else if(RadioGroup4->ItemIndex==3)
        pccCH=4; */

    ts=StrToFloat(Edit1->Text);
    prd=Edit5->Text.ToDouble();
    amp=Edit6->Text.ToDouble();
    offset=Edit7->Text.ToDouble();
    count=StrToInt(Edit8->Text);

```

```

//pci6204_set(pccCH);          /* 2相パルス、4通倍 など設定 */
pci3523DA_set(DACH);
pci3523AD_set(chn);

ci=prd/ts+0.5;
cih=(ci+1)/2;
for (i=0;i<=count;i++) {
    j = i%ci;
    if ( j<cih )msu[i]=offset+amp;
        else    msu[i]=offset-amp;
}

for (i=0;i<=count;i++) {

    tim_sync();
    da_out(msu[i],DACH);          //D/A 出力
    ad_value[i]=ad_in(chn);      //A/D 入力

//enc1=pci_6204_get(pccCH);      //カウントパルス読み込み
//pcc_value[i]=(float)enc1;

}

da_out(0.0,DACH);              //D/A 出力

button = Application->MessageBox("Finish!! Graph is OK and Memo is cancel","Status", MB_OKCANCEL);

if (button == IDOK) {
    Graph->ShowModal();
}
if (button ==IDCANCEL) {
    Memo1->Lines->Add("AD DATA");
    Memo1->Lines->Add(" ");
    //Memo1->Text="";
    for (i=1;i<=count;i++)
        Memo1->Lines->Add(FloatToStr(ad_value[i]));
}

}
//-----

void __fastcall TForm1::Button6Click(TObject *Sender)
{
    Graph->ShowModal();
}
//-----

void __fastcall TForm1::saveas1Click(TObject *Sender)
{
int i;
char *dataname;
FILE *fp;

    if (SaveDialog1->Execute() == true) {
        dataname=SaveDialog1->FileName.c_str();
        fp = fopen(dataname,"w");
        fprintf(fp,"%d¥n",count);

        for (i=0;i<count;i++)
            fprintf(fp,"%e %e¥n",msu[i],ad_value[i]);

        fclose(fp);
    }

}
//-----

void __fastcall TForm1::exit1Click(TObject *Sender)
{
pci_close(); //全ボード CLOSE
}

```

```

Close();
}
//-----

void __fastcall TForm1::Button7Click(TObject *Sender)
{
    pci_close(); //全ボード CLOSE
    Close();
}
//-----unit1.cpp 終了

```

```

//-----pci.h
#ifndef PCIH
#define PCIH
//-----
void pci_open(void); // 3個のボードを同時オープン
void pci_close(void); // 3個のボードを同時クローズ
void da_error_check(DWORD error); // 4ch DAボードエラーステータス表示
void ad_error_check(DWORD error); // 8ch ADボードエラーステータス表示
void timer_error_check(DWORD error); // タイマボードエラーステータス表示
void penc_error_check(DWORD error); // 2ch エンコーダカウンタボードエラーステータス表示

void pci3523DA_set(int ch); // DAチャンネル設定、レンジは+/- 10Vのみ
void da_out(float,int); // DA出力

void pci3523AD_set(int ch); // ADチャンネル設定、レンジは+/- 10Vのみ
float ad_in(int ch); // AD変換入力
void psignal (int cs1, int lp, float uu1[], float ts, float amp1,
             float ofst1, float prd1); // 波形生成
void tim_sync(void); // タイマ同期 新
//void tim_sync(float smp_prd); // タイマ同期
void pci6204_set(int ch); // エンコーダカウンタ、2相パルス4通倍
long pci6204_get(int ch); // エンコーダカウンタ値読み
//endif

```

```

//-----pci.cpp
#include <vcl.h>
#pragma hdrstop
//プライベート マクロ定義
#define PI 3.1416
#include <math.h>
#include "PCI.h"
#include "Unit1.h"

#include "FbiDa.h"
#include "FbiAd.h"
#include "Gpcda.h"
#include "Gpcad.h"
#include "Fbiwtim.h"
#include "FbiPenc.h"

#define PI 3.141592653589793238 //円周率

//-----
#pragma package(smart_init)
//-----
//プライベート変数
static HANDLE hDeviceHandle1, // デバイスハンドル(DA)
             hDeviceHandle2, // デバイスハンドル(AD)
             hDeviceHandle3, // デバイスハンドル(Timer)
             hDeviceHandle4; // デバイスハンドル(EncCounter)
static ADBOARDSPEC BoardSpec; // ADBOARDSPEC 構造体
static ADSMPLCHREQ SmpIChReq[8]; // ADSMPLCHREQ 構造体
static DASMPLCHREQ DaSmpIChReq[4];
static ULONG ulSmpIBufferSize; // アナログ出力バッファサイズ

```

```

//-----
void pci_open(void){
int error_flag=0;
// デバイス名"FBIDA1"のアナログ出力ボードの初期化
hDeviceHandle1 = DaOpen("FBIDA1");
if (hDeviceHandle1 == INVALID_HANDLE_VALUE) {
    Form1->Memo1->Lines->Add("error_DAOpen");
    error_flag=1;
}
// デバイス名"FBIA1"のアナログ入力ボードの初期化
hDeviceHandle2 = AdOpen("FBIA1");
if (hDeviceHandle2 == INVALID_HANDLE_VALUE) {
    Form1->Memo1->Lines->Add("error_ADOpen");
    error_flag=1;
}

// タイマボードの初期化を行います
//hDeviceHandle3 = TimerOpen("FBIWTIM", 0);
//if (hDeviceHandle3 == INVALID_HANDLE_VALUE) {
//    Form1->Memo1->Lines->Add("error_TimerOpen");
//    error_flag=1;
//}

// デバイス名"FBIPENC1"のエンコーダカウンタボードの初期化を行います。
hDeviceHandle4 = PencOpen("FBIPENC1", 0);
if (hDeviceHandle4 == INVALID_HANDLE_VALUE) {
    Form1->Memo1->Lines->Add("デバイス名 FBIPENC1 は使用できません");
    error_flag=1;
}
if (error_flag == 0) Form1->Memo1->Lines->Add("BOARD_OPEN_SUCCESS");
}

```

```

//-----

```

```

void pci_close(void){
// アナログ出力ボードの終了処理
DaClose(hDeviceHandle1);
// アナログ入力ボードの終了処理
AdClose(hDeviceHandle2);
// タイマボードの終了処理
//TimerClose(hDeviceHandle3);
// エンコーダカウンタボードの終了処理
PencClose(hDeviceHandle4);
}

```

```

//-----

```

```

//同期の関数 新
void tim_sync() {
    unsigned long dwValue;
    unsigned int error;
    do {
        error = AdInputDI( hDeviceHandle2, &dwValue );
    } while( (dwValue & 0x0001) != 0 );          /*If High then Loop */

    do {
        error = AdInputDI( hDeviceHandle2, &dwValue );
    } while( (dwValue & 0x0001) == 0 );          /*If Low then Loop */

    if(error != AD_ERROR_SUCCESS){
        Form1->Memo1->Lines->Add("AdInputDI");
        ad_error_check(error);
    }
}

```

```

//-----

```

```

void pci3523DA_set(int ch){
//    DaSmpIChReq[ch-1].ulChNo = ch;    //DA チャンネルセット
//    DaSmpIChReq[ch-1].ulChNo = ch;    //DA チャンネルセット
//    DaSmpIChReq[ch-1].ulRange = DA_10V; //レンジは±10V だけ
}

```

```

//-----

```

```

void pci3523AD_set(int ch){

```

```

    DWORD error;          // 関数の実行結果
//  SmpIChReq[ch-1].ulChNo = ch; //AD チャンネルセット
    SmpIChReq[ch-1].ulChNo = ch; //AD チャンネルセット
    SmpIChReq[ch-1].ulRange = AD_10V; //レンジは±10V だけ

//デバイスの情報取得
    error = AdGetDeviceInfo( hDeviceHandle2, &BoardSpec );
    if(error != AD_ERROR_SUCCESS){
        Form1->Memo1->Lines->Add("AdGetDeviceInfo");
        ad_error_check(error);
    }
}
//-----
//D A 変換関数
void da_out(float vo,int ch)
{
    int error,mm;

    if(vo<-10) vo=-10;
    else if(vo>9.997) vo=9.997;

    mm=vo/10*2048+2048;

    error = DaOutputDA( hDeviceHandle1, 1, &DaSmpIChReq[ch-1], &mm);
    if(error != DA_ERROR_SUCCESS){
        Form1->Memo1->Lines->Add("DaOutputDA");
        da_error_check(error);
    }

    //return vo;
}
//-----
//A D 変換関数
float ad_in(int ch){
//  unsigned long error,adData[8];
    DWORD error;          // 関数の実行結果
    WORD adData[8];      // データ配列
    float dVolt;

    error = AdInputAD( hDeviceHandle2, 1, AD_INPUT_SINGLE, &SmpIChReq[ch-1], &adData[ch-1]);
    if(error != AD_ERROR_SUCCESS){
        Form1->Memo1->Lines->Add("AdInputAD");
        ad_error_check(error);
    }
//電圧算出 BoardSpec.ulResolution=12(ULONG)
    dVolt = 20/pow(2,BoardSpec.ulResolution)*adData[ch-1] - 10;
    return dVolt;
}
//-----
void pci6204_set(int ch){
//選択したチャンネルを位相差パルスカウンタ 入力、4 通倍、非同期クリア、
//アップカウンタ、一致検出無効、ソフトウェアラッチに設定します
    int error;
    error = PencSetMode( hDeviceHandle4, ch, 6, 0, 0, 0 );
    if(error != PENC_ERROR_SUCCESS){
        Form1->Memo1->Lines->Add("PencSetMode");
        penc_error_check(error);
    }
}
//-----
long pci6204_get(int ch){
    int error;
    long count_value;
    error = PencGetCounter( hDeviceHandle4, ch, &count_value );
    if(error != PENC_ERROR_SUCCESS){
        Form1->Memo1->Lines->Add("PencGetCounter");
        penc_error_check(error);
    }
    return count_value;
}
}

```



//-----